REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-04-

0294

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
31-05-2004	Final	01-06-2002 – 31-05-2004

4. TITLE AND SUBTITLE
User Configurable Incident Response Monitor

5a. CONTRACT NUMBER
F49620-02-C-0032

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Anderson, Chris
Barber, K. Suzanne
Graser, Thomas J.
Park, Jisun
Swenholt, Mark
Yu, David NL

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Laboratory for Intelligent Processes and Systems
Elec. and Computer Eng. Dept., Mail Code: C0803
University of Texas at Austin
201 E. 24th St., ACE 5.446
Austin, TX 78712-1084

ScenPro Inc.
101 W. Renner Rd., Suite 130
Richardson, TX 75082

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
4015 Wilson Boulevard, Room 713
Arlington, VA 22203-1954

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSORING/MONITORING AGENCY REPORT NUMBER

12. DISTRIBUTION AVAILABILITY STATEMENT
Approve for Public Release: Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This document is the final report for contract F49620-02-C-0032. The contractual requirement for this report is defined in Exhibit A, Reporting Requirements Under Contracts Issued by AFOSR, as cited by contract line item 0002 of the contract. This report covers the work performed over the period from June 1, 2002 (the date of contract award) through May 31, 2004 (the date of the contract termination). Its contents cover the highlights of the research and implementation effort related to the User Configurable Incident Response Monitor (UCIRM) application defined in the contractor's Technical Proposal. The UCIRM, designed and implemented jointly by ScenPro, Inc. and the Laboratory for Intelligent Processes and Systems at the University of Texas at Austin (UT:LIPS), provides decision support for military personnel responsible for monitoring military installations to determine if designated zones (i.e. predefined regions) have been subject to chemical/biological attack. Specifically, the UCIRM supports personnel with respect to the following two decision-making tasks: (i) Estimating how zone contamination will affect the tempo of base operations and (ii) Determining zone contamination with an assigned confidence level based on the trustworthiness of sensor readings and the sensors providing those readings.

15. SUBJECT TERMS
Decision support, Uncertainty in Artificial Intelligence, Chem/Bio Incident Response

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE	Unclassified Unlimited	51	Mark Swenholt
Unclassified	Unclassified	Unclassified			19b. TELEPONE NUMBER (*Include area code*) 972-437-5001

Abstract

This document is the final report for contract F49620-02-C-0032. The contractual requirement for this report is defined in Exhibit A, Reporting Requirements Under Contracts Issued by AFOSR, as cited by contract line item 0002 of the contract. This report covers the work performed over the period from June 1, 2002 (the date of contract award) through May 31, 2004 (the date of the contract termination). Its contents cover the highlights of the research and implementation effort related to the User Configurable Incident Response Monitor (UCIRM) application defined in the contractor's Technical Proposal. The UCIRM, designed and implemented jointly by ScenPro, Inc. and the Laboratory for Intelligent Processes and Systems at the University of Texas at Austin (UT:LIPS), provides decision support for military personnel responsible for monitoring military installations to determine if designated zones (i.e. predefined regions) have been subject to chemical/biological attack. Specifically, the UCIRM supports personnel with respect to the following two decision-making tasks: (i) Estimating how zone contamination will affect the tempo of base operations and (ii) Determining zone contamination with an assigned confidence level based on the trustworthiness of sensor readings and the sensors providing those readings.

Table of Contents

List of Figures

List of Tables

1. Summary

The objectives for this program were defined in the Phase II Technical Proposal, which outlined an approach to deliver advanced decision support for military personnel responsible for monitoring military installations to determine if designated zones (i.e. predefined regions) have been subject to chemical/biological attack. The primary deliverables consist of (i) the User Configurable Incident Response Monitor (UCIRM) application and (ii) dissemination of the underlying research in information trustworthiness assessment that provides the foundation for advanced decision support using the UCIRM.

The objectives assigned to ScenPro were the definition of the UCIRM architecture and implementation of a prototype UCIRM application that provides the functionality described in the technical proposal. The UCIRM application was designed such that outputs from UCIRM analysis can be displayed by the Digital Dashboard application developed by the Air Force Research Laboratory (AFRL) branch at Rome, New York for the Theater Battle Management Core Systems – Unit Level (TBMCS-UL) system. ScenPro was also responsible for the development of a model that represents the effects of an attack with Chemical or Biological weapons on an air base in the operational domain.

The Laboratory for Intelligent Processes and Systems at the University of Texas at Austin (UT:LIPS) focused on research that utilizes knowledge about the trustworthiness of information sources and the confidence in the information provided by those sources to determine an overall level of belief for information presented to decision makers with an associated degree of confidence. This research was incorporated into an information trustworthiness assessment capability to support analysis in the UCIRM.

The highlights of the work performed since commencement of this contract are as follows:

ScenPro, Inc.

- Developed an approach to relate the incident domain data from a chemical, biological, radiological, or nuclear (CBRN) attack to the operational domain based on the geographical location of critical airbase resources.
- Implemented an Effects Model that provides the Operational Domain assessment of the effects of an attack.
- Defined an approach to display summary results from the array of Portal Shield sensors at Osan Air Force Base in terms of the arbitrarily defined NBC zones at the base.
- Drafted a Scenario of Use for demonstrating UCIRM functionality. This scenario is included in this document in Section 2.2 derive the requirements for the application and define the roles and needs of the application's users.
- Defined the UCIRM system architecture, incorporating both ScenPro and UT:LIPS components. The underlying Dashboard database structure was used as a guide for the UCIRM application.
- Identified a candidate transition activity for the UCIRM application. Specifically, ScenPro will work with the Air Force Research Laboratory (AFRL) branch at Rome, NY to explore opportunities to participate in the Contamination Avoidance at Seaports of Debarkation (CASPOD) advanced concept technology demo (ACTD).

Laboratory for Intelligent Processes and Systems (UT:LIPS)

- Adapted information trustworthiness assessment research originally motivated by multi-agent system software to the UCIRM. A proof of concept version of this functionality has been developed and tested in the UCIRM as part of an Information Trustworthiness Assessment Facility (ITAF) component.
- Advanced state-of-the-art in research associated with trustworthiness assessment.

Each of these topics is discussed in more detail in Section 3.

2. Introduction

This section describes the military domain problem addressed by this effort and presents a scenario of use for the UCIRM to support decision makers in this domain.

2.1. Problem Statement

A key issue in assessing the effects of an attack that uses weapons of mass destruction (WMD) is that key attributes and parameters describing the event and the related response may not support users in an operational domain. Consider the example of a scenario that includes an attack on an Air Force base that involves the release of a chemical agent. After the agent is released, the information needed and used by the incident responders likely includes the following key domain attributes:

- Location and scale of the contaminated area.
- The scope of any decontamination effort in terms of manpower and decontamination agent resources required.
- The manpower and equipment resources required to satisfy physical security requirements during the response effort.

In the operational domain, the desired information may be limited to an assessment of the impact on the number of aircraft that an installation can support and operate over the anticipated duration of the event. The information needed by the personnel involved in the operations at the base is not the information used by the incident responders in their work. Instead, they need an assessment of the effects on the mission(s) assigned to the base due to the attack and the associated response effort, as shown in Figure 1. The UCIRM provides this assessment.

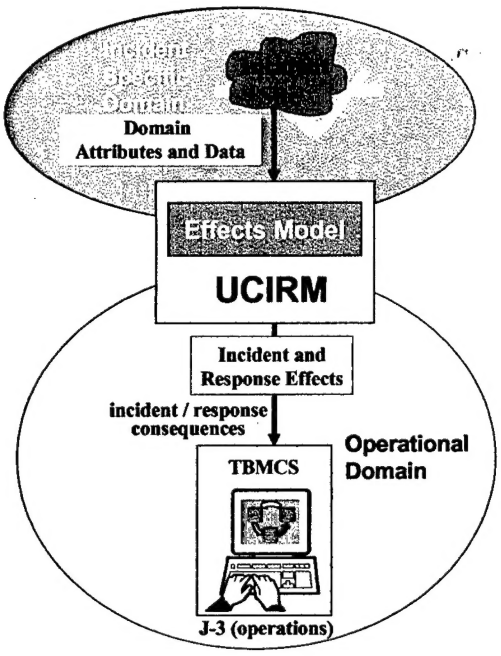

Figure 1: UCIRM Translates Information Between Domains

Figure 1 shows that the UCIRM provides a translation of incident specific domain data (for the event and its resulting response functions) to key parameters in the operational domain. The UCIRM accepts data from various sources in the incident domain to drive the effects model. The effects model provides the translation function and produces the effects of the incident and the response and the relevant consequences to the user in the operational domain.

The information provided by the UCIRM increases the effectiveness of personnel by indicating updated resource requirements and the consequences associated with incident response.

2.2. UCIRM Scenario of Use

ScenPro uses the Scenario-based Engineering Process (SEP) originally developed by the University of Texas at Arlington as the company's system development methodology. SEP emphasizes the use of scenarios to document and infer the user's requirements that will be met by the developed system or product. The following is the essential scenario of use developed for the Phase II UCIRM effort.

2.2.1 Event Summary

Recovery operations are underway in response to a wartime attack with chemical agents on Osan Air Force Base (AFB), located in South Korea.

2.2.2 Background

On the first day of a new war on the Korean Peninsula, Osan AFB personnel are working in a Restoration of Operations (RESTOPS) situation following an attack on the base by forces of the Democratic People's Republic of (North) Korea (DPRK). The attack was delivered by

several Scud-derivative (Hwasong-5 and Hwasong-6) tactical ballistic missiles (TBMs) that were armed with mustard chemical warheads.

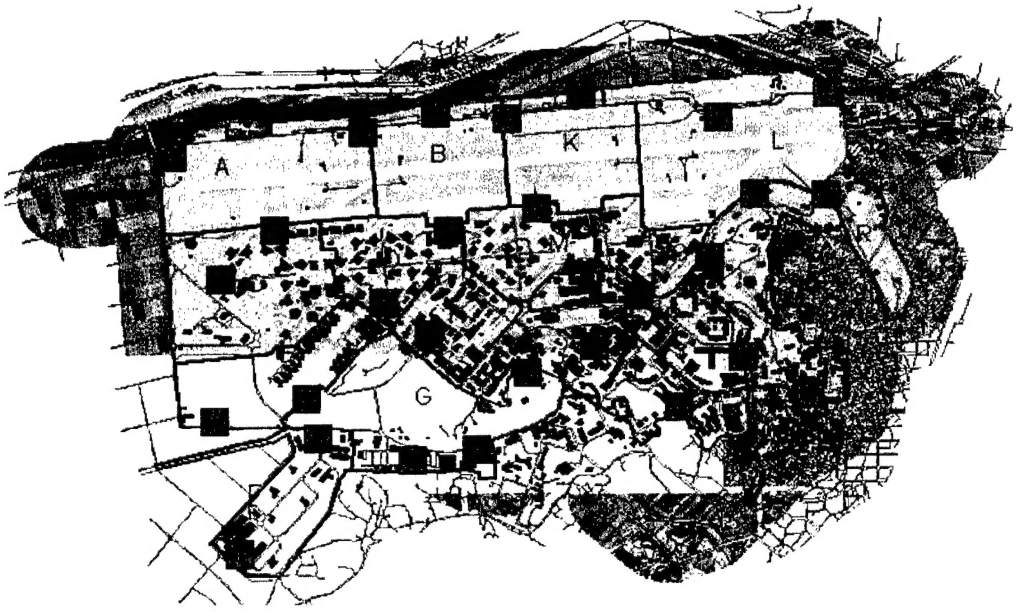

Figure 2: Osan AFB Map with NBC Zone Status Indications

Osan AFB has been tasked to supply aircraft to fly multiple missions defined by the current Air Tasking Order (ATO) in the near future. The operational tempo at the base is at a very high level due to the warning signs that an outbreak of war was imminent. The current aircraft assignments in the ATO were made the day prior to the attack, in accordance with the usual ATO procedure. The Wing Operations Center (WOC) staff is trying to provide a quantified assessment of the extent that the attack will affect the number of aircraft that the base can provide for the missions it's tasked to support. This assessment will be provided to the Joint Air Operations Center (JAOC) that manages the air campaign for the Korean Theater of Operations, and updated as soon as revised estimates become available.

2.2.3 Issues

- The indirect effects of the attack must be considered when estimating the impact of the attack on the number of A/C available. For instance, the effects of degraded performance of flight line personnel when wearing personal protective equipment (PPE) needs to be considered, not just the effects of personnel casualties and (unusable) contaminated ordnance and equipment.
- Information about detailed effects of attack will probably become available in bits and pieces, not as a whole, but assessments of the impact of the attack will be wanted as soon as possible. Accordingly, the assessment will be dynamic and time-variable.
- A quantified level of confidence in the assessment is desired. As more information regarding the detailed effects of the attack becomes available, the expressed confidence level of the assessment should increase.

2.2.4 Notional Scenario Initiation

1800 hours, the day prior to the attack (A-Day – 1): The Wing Operations Center (WOC) at Osan Air Force Base (AFB) in Korea receives their daily Air Tasking Order (ATO) for the

next day. At about the same time, the Strategic Rocket Forces of the DPRK receive their orders to launch an intensive attack on Osan AFB early the next morning to disrupt flight operations at the base during the first day of the war.

2.2.5 Sequence of Events

1800 – 2100, A-Day – 1: The Operations staff at Osan determines the specific aircraft to be used to support the ATO, and the service, personnel, and equipment that are needed to prepare them in the specified configuration. Aircraft assignments for Osan AFB in the ATO have been made on the basis of information about the resources (aircraft, consumables, personnel, equipment, and facilities) at Osan in the theater's Air Operations Database (AODB). As long as the data in the AODB is current, Osan should have the resources required to provide the number of assigned aircraft, in their designated configurations, specified in the ATO.

UCIRM Usage: The escalated level of tensions has led the Operations Group Commander in the WOC to direct his staff to prepare for an attack on the base. Accordingly, a member of the Operations Group staff uses the UCIRM to establish a baseline model of how the base uses assets and resources to provide the aircraft specified by the ATO. The UCIRM application accesses the data in the Wing Command and Control System (WCCS), Flight Operations (FLYOPS), and Global Combat Support System (GCSS) applications to create this model. The model includes an estimate of the likelihood that the base will be able to provide the aircraft as specified in the ATO. Barring an extraordinary event, of course, the value of this likelihood is very high. The Operations staff will use the UCIRM after the attack to assess deviations from this baseline model during the post-attack RESTOPS effort, and will update the estimate of the uncertainty of the model's results.

0300 – 0400, A-day: The base is attacked with a number of Scud-derivative TBMs armed with distilled mustard (HD). The persistence of the mustard agent has the potential to cause long-term degradation in base performance.

0315, A-day: The base commander activates the base's Survival Recovery Center (SRC). RESTOPS actions are deferred until the attack is over and an all-clear indication is received.

0420, A-day: The base command post orders that the all-clear signal be communicated. RESTOPS efforts begin, directed by the SRC staff.

0420-0830, A-day: Base staff completes the incident response tasks defined for the notification and initial response phases. The base commander requests that his staff provide periodic briefings that contain the following:

- a summary of the ongoing efforts in response to the attack
- an assessment of the base's capability to support the various missions scheduled in the ATO during the duration of the RESTOPS effort

UCIRM Usage: A member of the Operations Group staff in the WOC uses the UCIRM to create estimates of the deviation from the results of the baseline model created earlier as information about the effects of the attack becomes available during the RESTOPS effort. The UCIRM application uses its information uncertainty function to provide estimates of the range of the number of aircraft that expected to be available to support each mission in the

ATO.

The UCIRM estimates are used in the assessments of the base's capability to support the missions scheduled in the ATO during the RESTOPS effort.

Data for the available assets and the assets used to equip the aircraft for the ATO-designated missions was saved in the UCIRM application when the baseline assessment was performed. Changes in available assets due to the result of the attack are read from updates to the original data sources (WCCS, FLYOPS, GCCS, etc.) where available. The effects of the attack can also be entered by accessing the UCIRM's copy of the asset data via its user interface and updating the data in the application if there is a problem in connecting to, or updating, the original data sources.

0830-2400, A-day: Base staff completes the incident response tasks defined for the follow-on response phase. The base commander requests that his staff continue the periodic briefings that contain the following:

- a summary of the ongoing efforts in response to the attack
- an assessment of the base's capability to support the various missions scheduled in the ATO during the duration of the RESTOPS effort

UCIRM Usage: A member of the Operations Group staff in the WOC continues to use the UCIRM to create estimates of the deviation from the results of the baseline model created earlier as information about the effects of the attack becomes available during the RESTOPS effort. The UCIRM application also provides information about the relative trustworthiness of information sources used by the application since the attack.

2.2.6 Termination

When the RESTOPS effort has been completed, the base is expected to be operating at a normal operational tempo. Consistency should be re-established across the TBMCS, WCCS, FLYOPS, and GCCS applications and their databases (such as the AODB). The UCIRM application is no longer required.

3. Methods, Assumptions, and Procedures

This section describes the User Configurable Incident Response Monitor (UCIRM) application. Section 3.1. summarizes the approach for the UCIRM effort, followed by detailed sections on the two primary UCIRM components: the Information Trustworthiness Assessment Facility (ITAF) and the Incident Response Monitor / Effects Model (IRM/EM).

3.1. Summary of Approach

3.1.1 Translating Incident Response Data into Meaningful Operational Information

Many Air Force installations across the globe utilize emergency response/consequence management software to capture incident response data. One recently developed application is the Theater Battle Management Core Systems – Unit Level (TBMCS-UL). This application was developed for the Restoration of Operations (RestOps) ACTD program which was performed at Osan Air Force Base (AFB). TBMCS-UL has a feature called the

Electronic Attack Report, which displays information regarding the presence of chemical or biological agent hazards at different locations on the base. The report obtains information from two different types of sources. One source is a set of automated biological agent detection sensors called Portal Shield. These sensors are deployed as an array across the entire extent of the base, as shown in Figure 3.

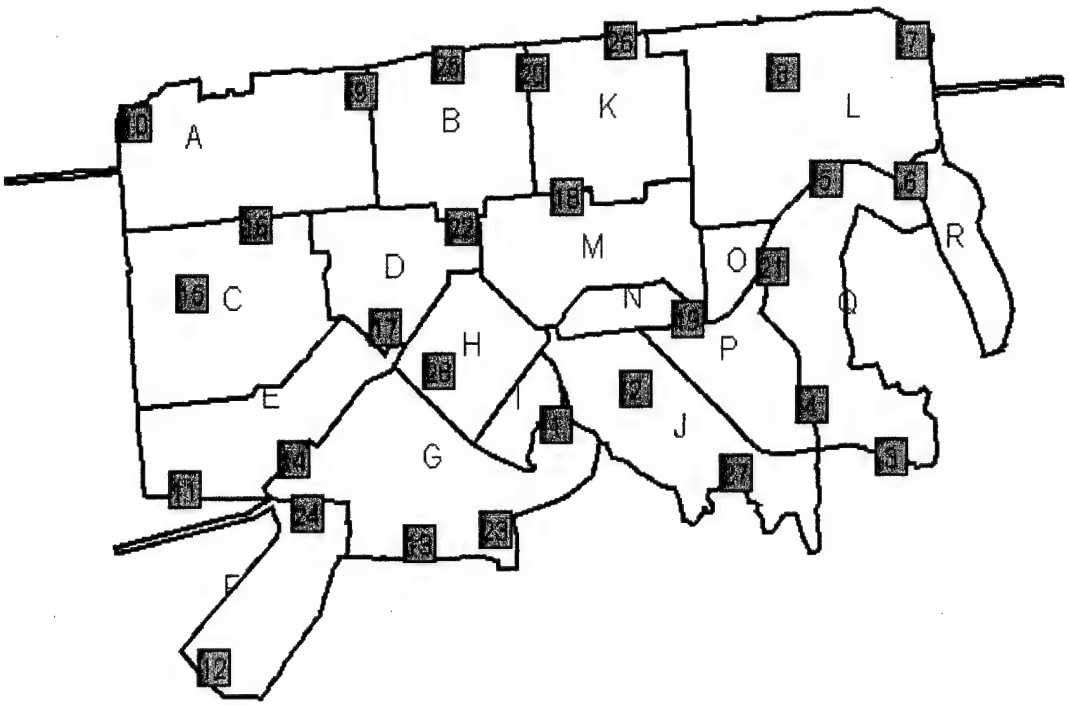

Figure 3: Portal Shield Sensor Locations at Osan AFB

The presence of chemical agents is currently obtained by deploying inspection teams with M8 Chemical Detector Paper. Each team makes two sweeps in the areas they are responsible for.

Osan AFB is divided into a set of 18 Nuclear, Biological, and Chemical (NBC) Zones that define areas on the base where personnel are required to wear PPE. The reported presence or absence of biological or chemical agents from the Portal Shield sensors and the results of the inspection teams regarding PPE requirements are made zone by zone. A representation of the Electronic Attack Report display that shows the status of the sensors and inspections is shown in Figure 4.

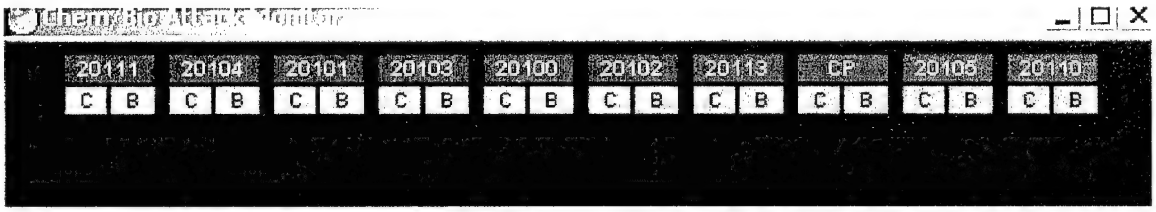

Figure 4: TBMCS-UL Electronic Attack Report Display

In developing the UCIRM, incident data in the TBMCS-UL database was extended to include a remote chemical agent alarm equivalent to what the Portal Shield sensors provide for biological agents. The characteristics of the M22 Automatic Chemical Agent Alarm (ACAD) were assumed for these postulated sensors, and it was assumed that they were placed at the same locations as the existing Portal Shield sensors.

The effects model allows the user to evaluate the effects of the incident in terms of additional time to complete the mission. The model also allows the user to apply different resources and priorities to explore ways to mitigate the impact or provide the correct capabilities for later missions. The ITAF application provides a tool to evaluation whether specific areas are contaminated and therefore impacted by the incident.

The UCIRM uses a representation of an Air Tasking Order (ATO) to define the missions assigned to an Air Force Base. The mission support model for such a base defines a relationship between the resources and facilities in the various areas of the base defined by the NBC zones and the assigned missions. The accessibility of resources and the efficiency of personnel working in contaminated areas are degraded, and the model evaluates whether this degradation impacts the base's ability to support the assigned missions.

The application incorporates an information trustworthiness assessment function in conjunction with the model to assist base staff in determining zone-by-zone contamination / alarm status. Zone status is assigned with a level of confidence based on relevant sensor input, as discussed in Section 3.1.2 .

3.1.2 Incorporating an Information Trustworthiness Assessment Capability

Military personnel monitoring air base operations rely on multiple sources for the detection of chemical/biological agents. Sources include HAZMAT personnel dispatched per pre-arranged inspection plans to automated sensors installed at fixed locations. These sources can be characterized by aspects such as the types of agents they are capable of detecting, the fidelity at which they can detect, the speed at which they operate, and their accuracy and reliability.

To make effective decisions regarding the safety of air base sectors and the performance of personnel operating in those sectors, monitoring personnel must cogently coalesce these information sources into a consistent set of "beliefs" regarding the presence of chemical or biological agents. The resulting set of beliefs provides a basis against which tactical decisions can be made. A key component of the coalescing process is maintaining a notion of "trust" regarding (i) the general reputation and capability of various detection approaches and technologies (based on accuracy, reliability, fidelity, etc.) and (ii) the confidence (i.e., estimated probability of a correctness) assigned to each detection reading received from a respective source.

To address the need for a rational coalescing process, the UCIRM effort leverages a state-of-the-art approach for "Trust Evaluation" originating from Artificial Intelligence (AI) research in Multi-Agent Systems (MAS) conducted by UT:LIPS. Intelligent software agents working in real-world domains often face partial, incomplete, uncertain, or even incorrect knowledge as it is provided by diverse information sources. Since agents in these domains do not have complete knowledge or ground truth information about the domain, each agent may hold different perspectives of environmental state. This assumption is true for many domains

where the global truth is not available or the cost of collecting and maintaining a centralized global perspective is prohibitive. As an agent builds its local perspective, the variance on the quality of the incoming information depends on the originating information sources.

Formalisms used for the maintenance of uncertain/incomplete information in such domains are often classified in two categories. In *logical (or symbolic) formalisms* (such as AGM revision [1], KM update [2], Iterated Revision (IR) [3][4], Transition-Based Update (TBU) [5], and Generalized Update (GU) [6]), mathematical logic is extended to represent intuitive patterns of reasoning that involve incomplete/uncertain information. In *numerical formalisms* (such as approaches based on probability [7] and Dempster-Shafer belief functions [8]), weights are associated with the pieces of knowledge, representing their degree of certainty.

Based on these formalisms, UT:LIPS developed a trust-based model for belief revision/situation assessment that integrates symbolic and numerical methods – using a symbolic method for consistency checking and a numerical method for credibility modeling. This approach takes advantage of the efficiencies gained by each formalism. The UT:LIPS situation assessment algorithm provides efficient numerical treatment of inherent uncertainty due to the nature of practical application domains by integrating existing techniques (e.g. Bayesian belief networks) [9].

This research has been implemented and verified within a software agent operating in a multi-agent environment, allowing the agent to assess the credibility of information provided by different sources. Equipped with the proposed model, an agent is capable of performing more sophisticated belief revision with better situation assessment in dynamic, uncertain, and even insecure environments. As a result, an agent using UT:LIPS trust-based belief revision exhibits cognitive capabilities that are (1) flexible and operational despite faulty information (e.g. due to faulty sensors or operator input, equipment performance, and communication links), and (2) capable of avoiding fraudulent information from unreliable or deceptive agents [10].

By leveraging trust-based belief revision approaches from UT:LIPS multi-agent research for the UCIRM effort, the UCIRM tool reuses a coalescing technology that has been empirically verified, thereby reducing development cost and improving product quality.

The successful execution of an ATO may be affected by numerous unexpected events, resulting in anticipated resources no longer being available and changes in the work environment that impede the performance of ATO-related tasks.

Ensuring that air base operations continue to run smoothly requires ongoing monitoring and adjustment by military personnel. The UCIRM provides a decision-support tool designed to aid monitoring for one such unexpected event: a chemical/biological attack. Such an attack could have a significant affect on ATO execution, not only rendering contaminated regions inaccessible, but also reducing the performance of ATO-related tasks due to requisite protective gear.

The UCIRM tool makes use of a novel combination of technologies to increase the effectiveness of military personnel monitoring for chemical/biological attacks and assessing their affect on ATO operations: (i) *trust evaluation:* trust evaluation is a natural approach for coalescing reports from the typical array of detection approaches employed (e.g., devices and

manual techniques having various fidelities); (ii) *geographical information system (GIS):* the UCIRM tool interfaces with the Digital Dashboard which provides airbase maps that offer monitoring personnel an intuitive means for assessing the contamination of different air base regions; and (iii) *expert system:* domain knowledge of chemical/biological agents and ATO-related tasks and resources is encoded in expert system rules, facilitating the cognition necessary to evaluate the extent of chemical/biological contamination and to determine the degree to which ATO operations are affected. Rather than attempting to completely replace monitoring personnel, the UCIRM tool is designed to augment decision-making as "computer-in-the-loop" support. Any assessments made by the tool are provided to decision-makers as decision support aids, so that the ultimate decision regarding contamination level and impact on ATO operations remains with the user.

Figure 5 illustrates the flow of information and reasoning conducted in the UCIRM tool pertaining to contamination and ATO operations in a sample air base region entitled "Region X." The objective of this reasoning is to assess the contamination level in a given region and infer the level of protective gear required. The ability to enter a given region and the protective gear required affect a ground crew's ability to accomplish their jobs and therefore their ability to satisfy the assigned ATO.

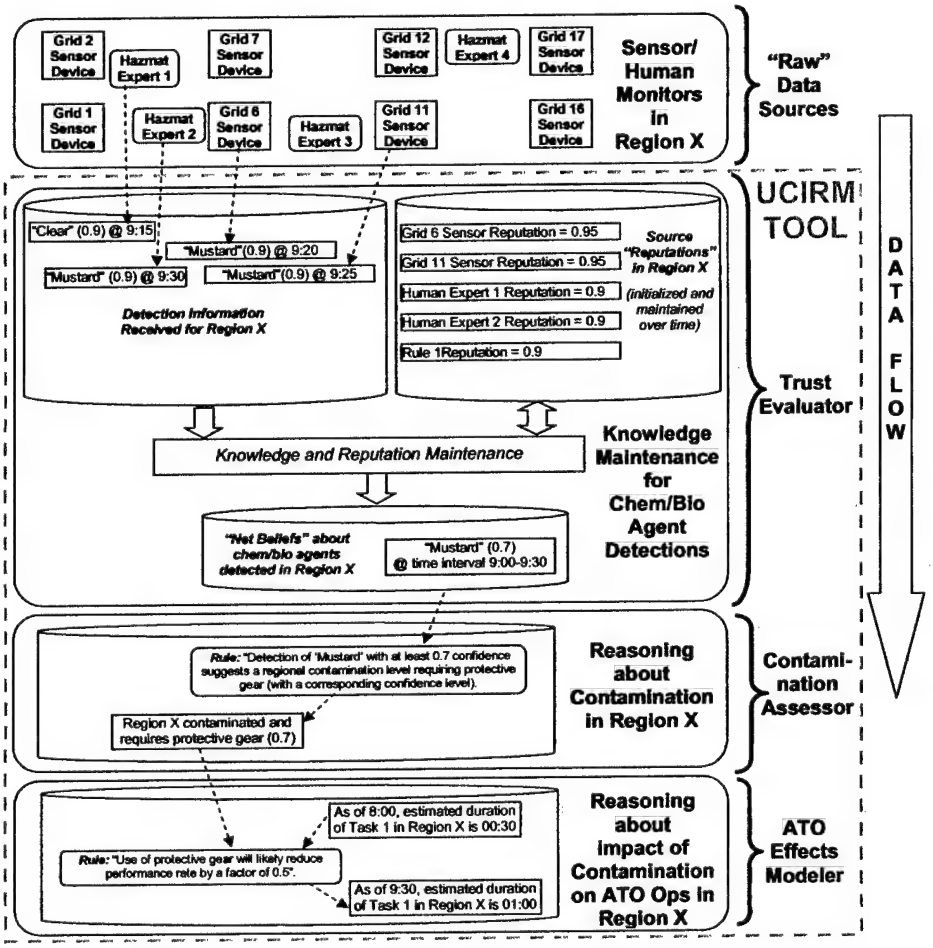

Figure 5: Trustworthiness Assessment Data Flow and Reasoning

An air base region is divided into roughly equal-sized grid entries, each associated with one or more sensor devices capable of detecting one or more chemical/biological agents at some level of fidelity. Figure 5 depicts sensor devices in grids 1, 2, 6, 7, 11, 12, 16, and 17 of Region X, as well as four HAZMAT experts dispatched to various locations within the region.

Reports from these sensors as well as samples taken and analyzed by HAZMAT personnel become the "raw data" used by the UCIRM tool. Each reading/report has an associated confidence (value from 0 to 1), assigned based on the accuracy of the sensor devices and the best judgment of the HAZMAT experts given the analysis method being used for a particular sample.

The data of interest in Figure 5 include:

- Negative ("Clear") reading from Hazmat Expert 1 recorded at 9:15 with a confidence of 0.9 when attempting to detect for mustard gas.
- Positive reading for mustard gas from Hazmat Expert 2 recorded at 9:30 with a confidence of 0.9.
- Positive reading for mustard gas by sensor device in Grid 6 recorded at 9:20 with a confidence of 0.9.
- Positive reading for mustard gas by sensor device in Grid 11 recorded at 9:25 with a confidence of 0.9.

While each of these reports was recorded at a specific time, decision-makers monitoring air base regions assess region contamination with respect to a selected time interval, which in this example can be considered to be 9:00 to 9:30. To effectively coalesce the raw data from sensors and HAZMAT experts, monitoring personnel must contend with disagreement among sensors on the presence of a chemical/biological agent – a negative reading of clear vs. positive readings of mustard gas. The tool's "Trust Evaluator" component is designed to address this issue. To arrive at a "net belief" regarding mustard gas contamination in Region X for the interval 9:00-9:30, the Trust Evaluator coalesces positive and negative readings, factoring in associated confidence values (0.9 for all readings in this example) and "reputations" for sensor devices and hazmat personnel (varying from 0.9 to 0.95 in Figure 5). These reputation factors are typically initialized based on characteristics such as device accuracy, HAZMAT personnel expertise, and device reliability and are adjusted as trust evaluation is conducted over multiple time steps. Since flagging a zone as contaminated has the potential to degrade human performance due to protective gear, decision makers want to avoid responding to false alarms. By factoring in sensor reputations and sensor reading confidence values when assessing zone status, decision makers are less likely to respond with a knee-jerk reaction, where a single positive reading causes the decision maker to mark a zone as contaminated (provided a zone is being monitored by more than one sensor).

Given a set of net beliefs regarding chemical/biological agents detected, the "Contamination Assessor" and "ATO Effects Modeler" are designed to assess the contamination of a respective region and determine the impact of that contamination on the execution of respective ATO tasks.

In the example in Figure 5, a Contamination Assessor rule suggests that the presence of mustard gas with a confidence of 0.7 is sufficiently severe to warrant labeling the respective region as "contaminated" with a confidence of 0.7, allowing the entry of base personnel into the region, but mandating their use of protective gear. Subsequently, the Effects Modeler assesses the impact of the "contamination" label on ATO operations with a rule suggesting that the latency for ATO-related "Task 1" will be increased from 30 minutes to 1 hour due to the need for protective gear.

3.1.3 UCIRM Interface with the AFRL / Rome Labs Digital Dashboard

The UCIRM uses the mapping function in the Digital Dashboard to display its zone-by-zone assessment of contamination status. To realize this, the UCIRM application stores its results in a database table used by the Digital Dashboard to display the status of the NBC Zones. A screenshot of the Digital Dashboard application is shown in Figure 6. This view shows a map of Osan Air Force base (AFB) with its NBC zones outlined. Zones are colored according to their contamination status and the level of MOPP gear required.

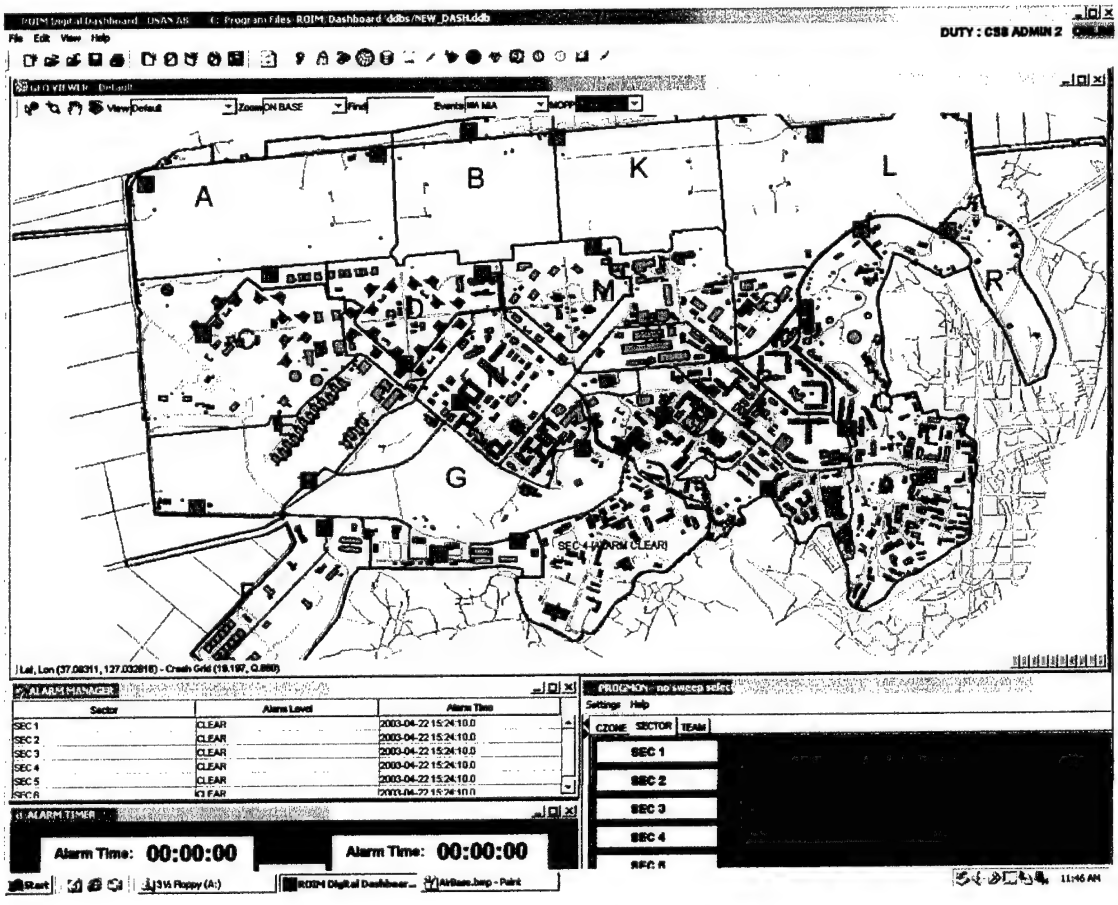

Figure 6: Map of Osan AFB in the Digital Dashboard Application

3.2. Information Trustworthiness Assessment Facility

This section discusses topics regarding the Information Trust Assessment Facility (ITAF), a distinguishing component of the UCIRM application. The following subsections address underlying research in trustworthiness assessment and implementation of the ITAF component.

3.2.1 Theory Underlying the ITAF

This section describes theoretical aspects related to trustworthiness assessment and its impact on decision support.

3.2.1.1 Applying Trustworthiness Assessment in Agent-based Software Systems

Supporting Decision Making by Assessing Trustworthiness when Maintaining Beliefs in Uncertain Environments

Decision-makers in domains such as chemical/biological contamination assessment must often render decisions based on incomplete and/or uncertain knowledge. Applications such as the UCIRM must provide features for managing this uncertainty. The ITAF provides this capability, and the research underlying the ITAF is based on prior and current work in UT:LIPS in the discipline of Uncertainty in Artificial Intelligence (UAI), a major sub-discipline of AI.

This research is applicable to multi-agent systems, where agents perceive their environment and establish their beliefs based on multiple information sources. The research is also applicable in domains where at least one entity is gathering information from other entities; the environment is dynamic and uncertain; and/or the information recipient's decisions or actions rely on the quality of information provided, requiring that the recipient be capable of filtering unreliable information.

Agents in open multi-agent systems (open MAS) often collect partial, incomplete, uncertain, or even incorrect knowledge from diverse information sources. Incorporating reliability information into belief revision mechanisms is essential for agents in real world multi-agent systems not only to resolve conflicts between incoming information, but also to model reliability or trustworthiness of the sources of information.

This approach to multi-agent belief revision combines symbolic and numerical operations. Specifically, it employs operations styled after an Assumption-based Truth Maintenance System to handle symbolic information, and Bayesian Network theory to treat numerical data. It distinguishes two knowledge bases: the knowledge background (*KB*), which is the set of all the pieces of knowledge available to the reasoning agent (since it can be inconsistent, it cannot be used in whole to support reasoning and decision); and, the knowledge base *K*, which is the maximally consistent, currently preferred piece of knowledge that should be used for reasoning and decision support. The incoming information, with its weight of evidence, is integrated not just into *K*, but also into the overall *KB*.

The belief revision process is computationally very expensive. From an engineering point of view, this means it is not always convenient or possible to build systems in which the belief revision process is performed only by a central unit. This motivates a distributed approach,

where the belief revision process is assigned to a society of agents, in which each agent is capable of performing local belief revision and communicating with others. The main difference in the distributed approach is that there is no longer a single agent with a global view of the system; each agent has a partial view. This allows the computational load to be divided among agents participating in the group.

An agent learns the trustworthiness of other agents and information sources using (1) dissimilarity measures calculated during belief revision invocations from previous time steps (Direct Trust Revision), (2) communicated trust information from other agents (Indirect Trust Revision), or (3) combinations of the previous two approaches (Hybrid Trust Revision). The algorithm proposed in this research mandates that agents with consistently low reputations will eventually be isolated from the agent society since other agents will rarely accept justifications or arguments from agents with relatively low reputations. This isolation of unreliable information sources from the system is called "soft security [10]," in contrast to hard security (infrastructure level security such as secure communication with public keys or authenticated name services).

A Framework for Belief Revision (Maintenance) in an open Multi-Agent System (MAS)

To implement Assumption-based Truth Maintenance system, the sentence-based (i.e., knowledge propositions) approach to the revision of a cognitive state involves two knowledge repositories:

- Knowledge Background KB is the set of all the propositions available to the reasoning agent; it may contain all the information received by the agent during its cognitive life; KB may be inconsistent;
- Knowledge Base $K \in KB$, which is the maximal consistent subset of KB, currently preferred pieces of knowledge that should be used for reasoning and decision support.

Given incoming information q, computationally, the overall belief revision approach consists of four steps [10].

Step 1: generation of all possible worlds (the maximal consistent subsets of KB)
Step 2: revision of the credibility weighs of the sentences in KB (through forward message passing in the Bayesian network) and creation of KB'
Step 3: selection of a preferred maximal consistent subset of KB as the new revised base K'
Step 4: creation of K^*, the logical expansion of K'

The incoming information (knowledge) q, with its weight of evidence, is confronted not just with the current base K, but with the overall knowledge background KB, so that the degrees of credibility of the sentences in $KB \cup \{q\}$ are reviewed on a broader and less prejudicial basis (Step 2). The main advantage is that we can recover discarded sentences from KB for the maximal consistency of K' (Step 3). If only K is revised by q then information cannot be recovered from KB. Step 3 might select a new base K' to be the same as the previous K (meaning that q has been rejected) but, in general, K' will have a different credibility

distribution than that of K. q might be rejected even if Step 3 chooses a base K' different from K, but still contains sentences incompatible with q. Even if q is consistent with K, $K' = KB \cup \{q\}$ does not necessarily hold true, since Step 2 may yield a totally different choice at Step 3 due to different distributions of reputation or confidence values. Previously rejected pieces of knowledge M ($M \subset KB$) can be rescued simply by determining some upsetting between the credibility of a set N ($N \subset KB$) and the credibility of M. This may happen if q supports M against N.

Step 1 deals with consistency and derivation, and acts on the symbolic part of the information. Operations are in ATMS style (Assumption-based Truth Maintenance System) to find all possible worlds using a set-covering algorithm [11].
In Step 2, an agent assigns revised credibilities on all sentences in *KB* based on singly-connected Bayesian networks. These networks are formed with incoming information and the current reputations of respective information sources. Step 2 consists of the following sub-steps.

(Sub-Step 2.1) Poly-tree constructions based on information sources' justification on knowledge q: The agent builds inference poly-trees, singly connected *Directed Acyclic Graphs* (DAG), for each sentence in *KB* from the justifications accumulated for the given knowledge q.

(Sub-Step 2.2) Belief revision on the certainty of knowledge q: The agent revises certainty factors of the sentences (measures that present how believable a sentence is) in *KB* by combining evidence and updating belief nodes in the given poly-trees.
Once the poly tree is built for incoming information q in Sub-Step 2.1, the certainty value Agent X has on q, P(q=*true*) or simply P(q^T) can be calculated by propagating probabilities in the tree. The numerical treatment here follows the algorithm developed by Neapolitan [12], poly-tree probability propagation. The poly-tree probability propagation is an algorithm that updates probabilities incrementally as information becomes available. Following Neapolitan [12] and assuming that the reliability of information sources is conditionally independent, probability propagation downward from information sources to q is given by[1]

$$\pi(q^T) = \sum_{a_1=\{T,U\}} \cdots \sum_{a_k=\{T,U\}} P(q^T \mid S_1^{a_1}, S_2^{a_2}, \ldots, S_k^{a_k}) \pi(S_1^{a_1}) \pi(S_2^{a_2}) \ldots \pi(S_k^{a_k})$$

$$= \sum_{a_1=\{T,U\}} \cdots \sum_{a_k=\{T,U\}} P(q^T \mid S_1^{a_1}) P(q^T \mid S_2^{a_2}) \ldots P(q^T \mid S_k^{a_k}) \pi(S_1^{a_1}) \pi(S_2^{a_2}) \ldots \pi(S_k^{a_k})$$

$$\pi(q^F) = \sum_{a_1=\{T,U\}} \cdots \sum_{a_k=\{T,U\}} P(q^F \mid S_1^{a_1}) P(q^F \mid S_2^{a_2}) \ldots P(q^F \mid S_k^{a_k}) \pi(S_1^{a_1}) \pi(S_2^{a_2}) \ldots \pi(S_k^{a_k})$$

$$\begin{cases} \pi(S_m^T) = P(S_m^T) = \{\text{current reputatoin of } S_m\} \\ \pi(S_m^U) = 1 - \pi(S_m^T) \\ P(q^T \mid S_1^T) = \{\text{certainty values given by } S_1\} \end{cases}$$

with 2^k terms where

[1] In this context π represents the probability propagation function.

$$q^T \equiv (q = true),\ q^F \equiv (q = false),\ S^T \equiv (S = Trustworthy),\ \text{and}\ S^U \equiv (S = Untrustworthy)$$

In Step 3, once all belief beliefs in the *KB* are assigned revised credibility, one of possible worlds (K_i) generated in Step 1 is selected as a preferred maximally consistent knowledge base K'. Among many possible worlds K_i, the one with the greatest average credibility (in other words, most probable in general) will be awarded as K'. The algorithm is formulated as follows.

For all sentences in a possible world K_i, find K' such that

$$\arg\max_{K' \in \{K_i\}} \left\{ \frac{\sum_{c=1}^{N} P(q_c = true)}{N} \right\}$$

where N is total number of sentences in a K_i, and q_c is a sentence in the K_i.
Step 4 is not particularly significant since, theoretically, it simply consists of applying classic logical entailment on the preferred sentences (sentences in K') to deduce a plausible conclusion from it. This last step selects from the derived sentences, all those whose origin set is a subset of the preferred sentences.

The Impact of Trustworthiness Assessment on Agent Autonomy

A characterizing feature of agent-based systems is the notion of individual agent autonomy – the degree to which an agent perceives, reasons, and acts independently to achieve its goals. The goals and beliefs of an agent are not independent of each other. The agent's beliefs about itself, others and the environment are based on its derived models of perceived and communicated information. In order for an agent to be autonomous, an agent must control its beliefs as well as tasks/goals. This research asserts that the degree of belief autonomy reflects the degree of control an agent maintains over its beliefs. In other words, belief autonomy is defined as an agent's degree of dependence on external information sources to form its beliefs. This implementation is based on the premise that an agent should select an appropriate degree of belief autonomy for a respective goal using the following factors: the information source trustworthiness, coverage and cost.

Trustworthiness of an agent can be represented by the notion of reputation which is a probability that the information provided by the agent is true. Therefore, trustworthiness is an expression for the reliability of the information sources. Coverage is a measure for an information source's contribution to an agent's information needs. The cost of getting information from a source is defined by the timeliness of information delivery from the source.

Since each agent only knows about a limited amount of information sources, it must also rely on other agents to share their models describing sources they know about. The degree of the model sharing refers to the degree to which a respective agent shares its knowledge about its "neighbors" (i.e. shares trustworthiness, coverage and cost data regarding information sources it knows about).

In order for an agent to determine its most appropriate degree of belief autonomy for a goal, it must determine (1) the potential sources that might deliver information it needs given information sources it knows about as well as sources discovered through model sharing and (2) the quality and efficiency of those sources by evaluating a source's trustworthiness, coverage and cost. A respective agent evaluates the discovered sources with regard to those metrics, and selects the most appropriate sources from those potential sources. The dependences on the selected sources are controlled by an agent itself (i.e. an agent determines from whom it will gather information to form its beliefs); thus, an agent controls its degree of belief autonomy.

3.2.1.2 Experiments Demonstrating the Benefits of Trust Assessment

The fundamental research hypothesis to be tested in empirical analysis of this approach is as follows:

> *Maintaining a memory of acquired knowledge quality and knowledge source reliability yields an improved belief maintenance process.*

The goal of this set of experiments is to provide motivation for trust modeling by showing empirical data describing how an agent performs better with perfect reputation information against belief revision without trust modeling. The setup involves "disease monitoring agents," decision makers assessing the severity of disease outbreak in a given region based on diagnoses, and "case agents," sources of diagnosis information.

The following assumptions are used throughout this set of experiments:

- Each disease monitoring agent knows about precise (ground-truth) reputation values of all agents in the system;
- Each case agent sends individual patient diagnoses and is not aware of ground truth diagnostics; and
- Communication channels are always available.

For these experiments, the following table shows control variables and respective ranges.

Control variable	Range
Number of disease agents	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
Number of unreliable agents	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
Level of uncertainty	0 ~ 1
Number of diseases	3, { anthrax, flu, Sudden Acute Respiratory Syndrome (SARS) }
Number of runs	1000 (per configuration)
Average reputation of unreliable agents	0.3
Average reputation of reliable agents	0.8

Table 1: Control Variables and Respective Ranges for Experiment Set 1

Results of Experiment Set 1 were measured in terms of Miss Rate (error), calculated as the frequency that the disease monitoring agent identified the correct patient disease. For example, if a disease monitoring agent correctly identifies a patient's disease 6 times out of 10, the miss rate is 1 - 6/10 = 0.4. The lower the miss rate, the more accurate the algorithm is.

Figure 7 and Figure 8 exhibit the performance of the belief revision algorithm as the total number of agents (x axis) and the percentage of unreliable agents (y axis) are varied. While these figures allow us to examine the dynamics of both algorithms under different configurations, the graphs are easily compared.

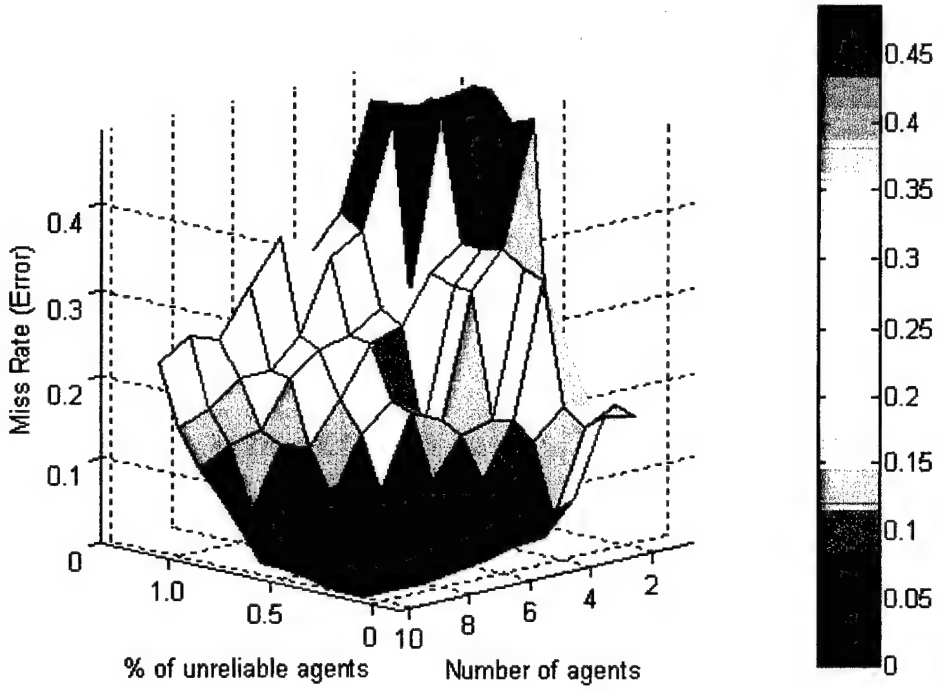

Figure 7: Accuracy of Belief Revision without Trust Information

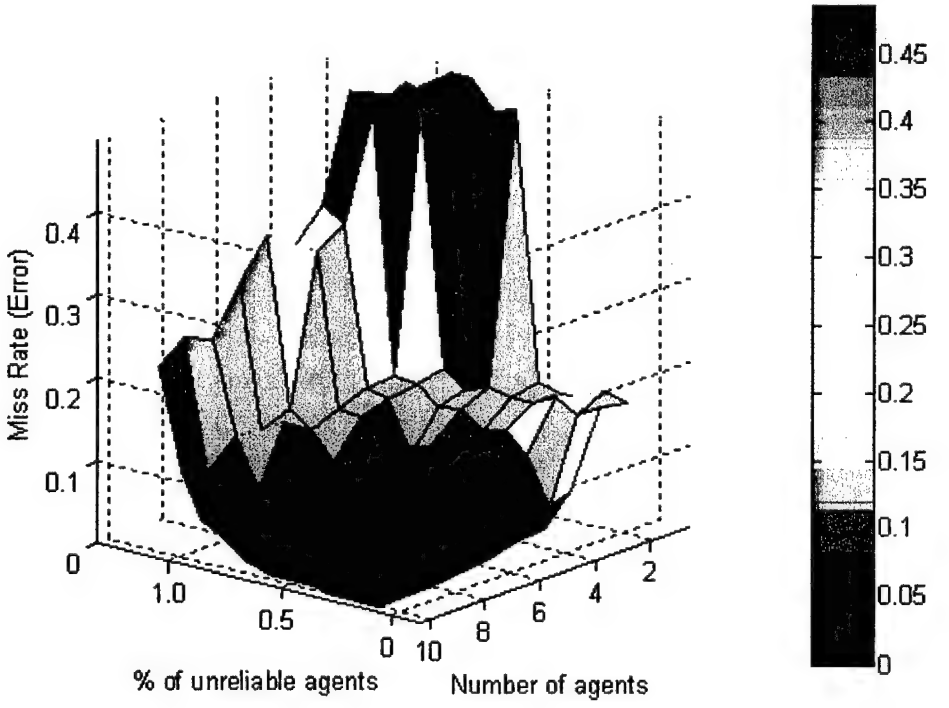

Figure 8: Accuracy of Belief Revision with Trust Information

These three-dimensional diagrams, which show the results of multiple runs of the ITAF algorithm, display error rates for given combinations of inaccurate sensors (unreliable agents) to the total number of sensors (agents). The key difference between them is in the reduction of error rates both as erroneous sensors decline, which is to be expected, and the fact that the more sensors in the network the less impact erroneous sensors have on the system as a whole. This is shown by the increased number of low error rate combinations seen in Figure 8. To compare performance of the belief revision algorithm with and without trust information, Figure 7 and Figure 8 are dissected along the x and y axes, creating Figure 9 and Figure 10, respectively.

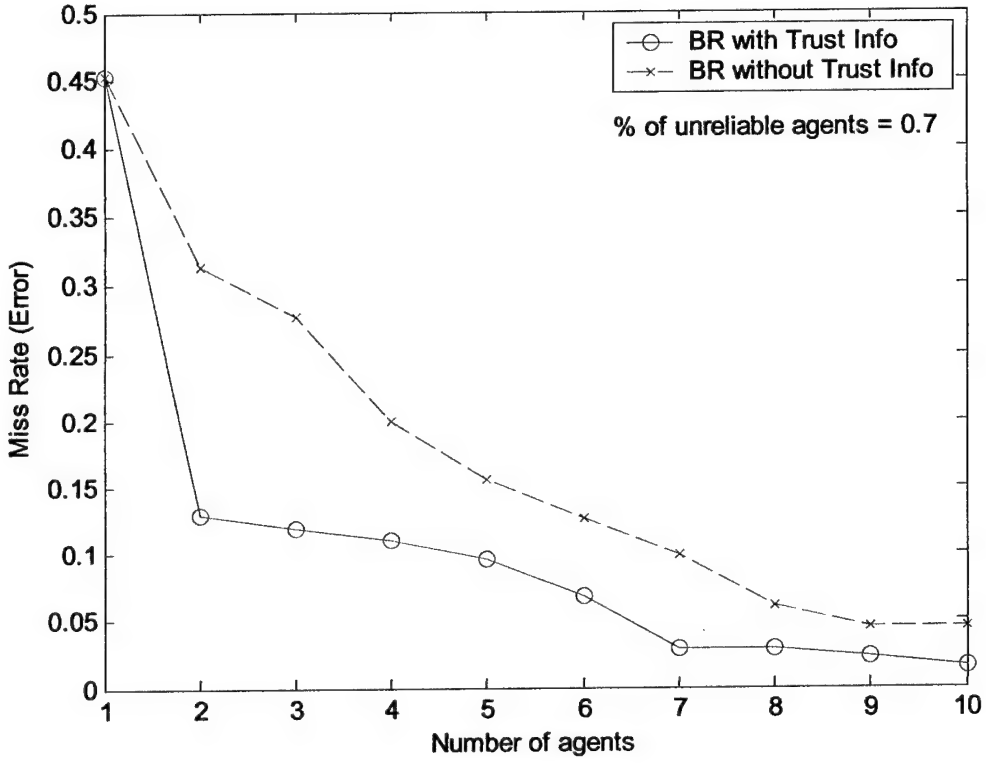

Figure 9: Miss Rate When The Number Of Agents Increases

Figure 9 shows that both algorithms generate fewer errors as the number of agents increases from 1 to 10 (when 70% of the agents are unreliable). This figure motivates incorporating trust information into the belief revision process since it displays:

- Significantly lower miss rate (significant with 95% confidence) with same number of agents and
- Faster convergence of errors.

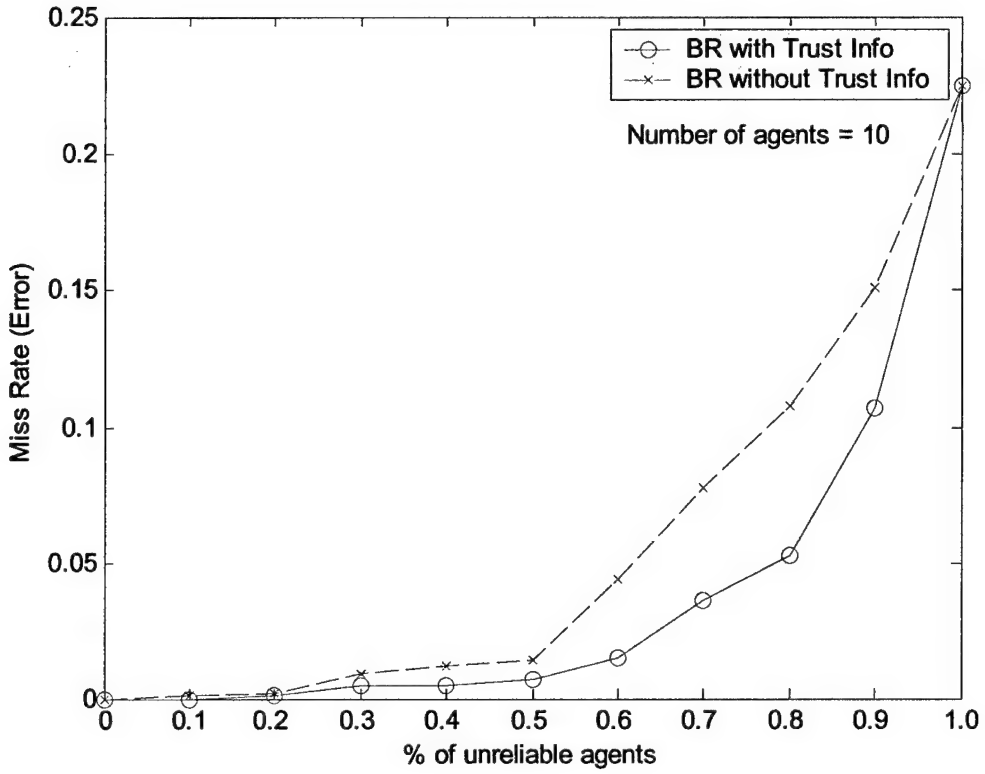

Figure 10: Miss Rate When Number Of Reliable Sources Increase

Figure 10 shows the dynamics of both algorithms as the ratio of unreliable agents increases from zero percent to 100 percent. The error increases due to incorrect information from unreliable agents. Trust information also helps in the face of an increasing number of unreliable agents. However, as shown in this figure, there is not significant improvement when every agent is trustworthy (meaning they are sending quality information) or every agent is unreliable.

3.2.2 ITAF Implementation

This section describes the ITAF implementation that demonstrates the value of utilizing trustworthiness assessment to help determine the contamination status of airbase chemical zones, where multiple chemical and biological sensors report detections for each zone.

3.2.2.3 ITAF Application Architecture

The ITAF applies trustworthiness assessment to manage data and data sources about chemical/biological contamination status for areas in the region of interest (zones) on behalf of the UCIRM Incident Response Monitor / Effects Modeler (IRM/EM). While the ITAF was designed to operate standalone, it is effectively integrated with the Digital Dashboard and IRM/EM via data exchange through the Air Operations Database (AODB).

Figure 11 depicts the architecture for the ITAF application. Components include the ITAF Control UI (Java), the ITAF Trust Assessor (Java and Matlab), and the database (Oracle). The user interacts with the application through the ITAF Control UI, which allows the user to invoke the trust assessment function and marshals data between the database and the ITAF.

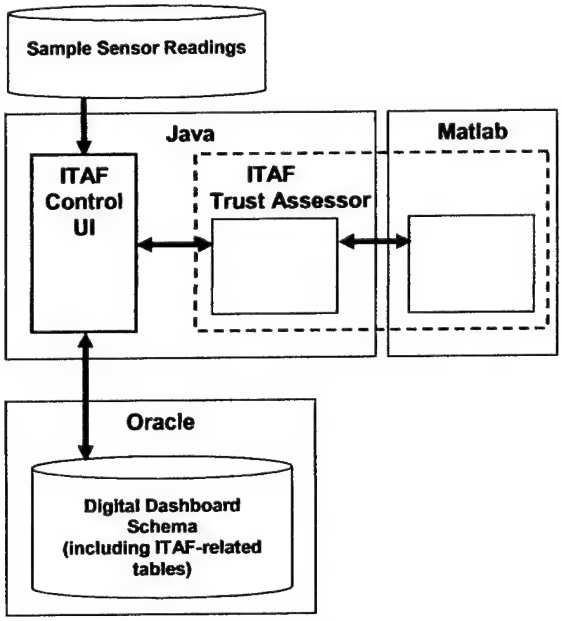

Figure 11: ITAF Prototype Architecture

In the ITAF application, sensor readings can be entered manually or imported from a text file, and trust assessment is invoked on demand, processing the data corresponding to a given time window. Figure 12 provides examples of three sensor readings for timestamp "12/19/02 08:02." Each reading includes the sensor identifier, a contamination status ("clear" or "alarm"), and a confidence on the status measurement between zero and one.

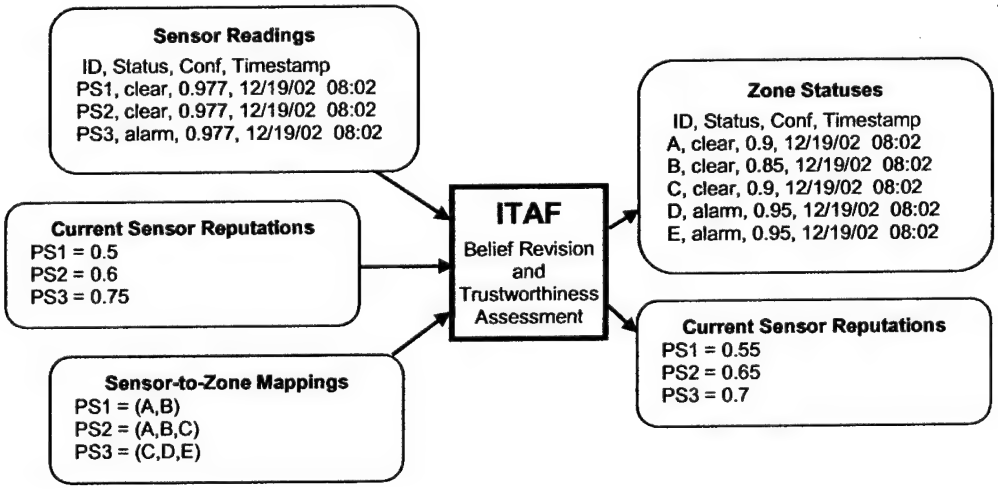

Figure 12: ITAF Belief Revision and Trustworthiness Assessment Inputs and Outputs

Each sensor is mapped to one or more zones on the airbase. ITAF considers (i) the current sensor reputations, (ii) the sensor reading values and confidences, and (iii) the sensor-to-zone mappings to determine a contamination belief for each zone and a degree of confidence in that belief. After belief maintenance and trust assessment, ITAF revises the respective sensor reputation values.

3.2.2.4 ITAF Application Design

Lessons learned from an ITAF proof of concept developed early in the program effort were considered in deriving the ITAF design for the UCIRM implementation. In particular, the proof of concept successfully demonstrated the interface between java and Matlab in a runtime environment, and experiences with the database tables created for use in the proof of concept motivated a more comprehensive schema design. The following describes the ITAF application design for the UCIRM.

Salient Requirements

The features included in this design reflect discussions between ScenPro and UT:LIPS as well as feedback from evaluation of the ITAF proof of concept developed by UT:LIPS. Specific requirements include:

- Ability to invoke ITAF belief maintenance and trust assessment with input consisting of sensor readings drawn from a "sliding time window." The size of the time window may change with different ITAF invocations.
- Support for persistent storage in AODB database of all historical information, including sensor readings received, sensor readings used for each belief maintenance invocation, zone contamination statuses produced by belief maintenance, and sensor reputations maintained by trust assessment. These data reflect "system knowledge" and allow ITAF input to be modified and trust assessment reprocessed for a given time window. Storage in the database allows easy access to data by the Effects Model and Digital Dashboard.
- Each sensor may have a different initial reputation.
- Ability to display all relevant information for each time window for which trust assessment was invoked, including the following:
    - Sensor inputs and confidences considered in the time window.
    - Zone statuses and confidences resulting from trust assessment.
    - Sensor reputations before and after trust assessment.

Data Definitions

Newly defined AODB database tables and respective columns include the following:

- ITAF_CONTROL – Miscellaneous parameters used by the ITAF process. Parameters include size of "sliding time window."

Column	Type	Description
TIME_WINDOW_SIZE	NUMBER	ITAF calculation time window

- ITAF_SENSOR – Sensor initial reputations.

Column	Type	Description
SENSOR_ID (KEY)	VARCHAR	Sensor ID
INITIAL_REPUTATION	NUMBER	Initial ITAF reputation

- ITAF_SENSOR_READING_INPUT – Readings as used for a given ITAF invocation (i.e., the collection of readings for an invocation time represent all readings associated with the defined time window). Allows a given window to be recomputed if necessary.

Column	Type	Description
ITAF_TIME_WINDOW_START (KEY)	DATETIME	Time window start time
ITAF_TIME_WINDOW_END(KEY)	DATETIME	Time window end time
SENSOR_ID (KEY)	VARCHAR	Sensor ID
READING_TIMESTAMP (KEY)	DATETIME	Reading (sensor report) time
READING_CLASS	VARCHAR	Type of chemical (e.g. nerve, blister, blood, pulmonary) or biological (e.g. toxin, viral, bacterial) agent detected
STATUS	VARCHAR	"ALARM" or "CLEAR"
CONFIDENCE	NUMBER	Confidence provided by sensor

- ITAF_SENSOR_READING_RECD – Readings as received from sensors or entered manually with corresponding reading timestamp. Readings with timestamps falling within a given time window are selected for trust assessment using that window (i.e. sensor readings are moved to ITAF_SENSOR_READING_INPUT for input to trust assessment if TIMESTAMP falls between ITAF_TIME_WINDOW_START and ITAF_TIME_WINDOW_END).

Column	Type	Description
SENSOR_ID (KEY)	VARCHAR	Sensor ID
READING_TIMESTAMP (KEY)	DATETIME	Reading (sensor report) time
READING_CLASS	VARCHAR	Type of chemical (e.g. nerve, blister, blood, pulmonary) or biological (e.g. toxin, viral, bacterial) agent detected
STATUS	VARCHAR	"ALARM" or "CLEAR"
CONFIDENCE	NUMBER	Confidence provided by sensor

- ITAF_SENSOR_REPUTATION – Sensor reputations for a given ITAF "time window" (identified by the window start time). Output from a respective ITAF invocation, and input to the following invocation.

Column	Type	Description
SENSOR_ID (KEY)	VARCHAR	Sensor ID
ITAF_TIME_WINDOW_START (KEY)	DATETIME	Time window start time
ITAF_TIME_WINDOW_END(KEY)	DATETIME	Time window end time
REPUTATION	NUMBER	Sensor reputation for ITAF

- ITAF_SENSOR_ZONE – Mapping between sensors and associated zones (with any degree of coverage). For detail on the algorithm used to associate sensors to respective zones, see Section 3.2.2.5.

Column	Type	Description
SENSOR_ID (KEY)	VARCHAR	Sensor ID
ZONE_ID (KEY)	VARCHAR	Zone ID

- ITAF_ZONE – Zones are defined (primarily for relational integrity).

Column	Type	Description
ZONE_ID (KEY)	VARCHAR	Sensor ID

- ITAF_ZONE_STATUS_OUTPUT – Net zone assessments from a given ITAF invocation.

Column	Type	Description
ITAF_TIME_WINDOW_START (KEY)	DATETIME	Time window start time
ITAF_TIME_WINDOW_END(KEY)	DATETIME	Time window end time
ZONE_ID (KEY)	VARCHAR	Zone ID
STATUS	VARCHAR	"ALARM" or "CLEAR"
CONFIDENCE	NUMBER	Confidence computed by ITAF from sensor reputations and confidences

Existing AODB database tables used by the Digital Dashboard that are also used by the ITAF include the following:

- CZONE – Contains chemical zone current MOPP level and Alarm/Clear status. The Digital Dashboard consults the CZONE table to determine how to render chemical zones on the airbase chemical zone map. Specifically, zones are colored according to assigned MOPP level.

Column	Type	Description
CZONE_ID	VARCHAR	Chemical zone ID
MOPP_CON_TYPE_ID	VARCHAR	MOPP level identifier (see MOPP_CON_TYPE table below)
ALARM_TYPE_ID	VARCHAR	Textual description of alarm status (e.g. "RED" for alarm vs. "GREEN" for clear)
CZONE_CLEAR_CD	NUMERIC	Numeric alarm indicator (1=Alarm, 0=Clear)
CONFIDENCE	NUMERIC	Confidence 0-1 assigned by ITAF to current zone status (newly added field to accommodate ITAF)

When the ITAF determines that a zone is contaminated (i.e. has status "Alarm") with a given confidence based on sensor inputs, the ITAF updates the respective row in the CZONE table, setting the MOPP_CON_TYPE_ID based on the status confidence (see below), setting CZONE_CLEAR_CD to 1, and setting the ALARM_TYPE_ID to "RED".

When the ITAF determines that a zone is clear from contamination (i.e. has status "Clear"), the ITAF updates the respective row in the CZONE table, setting CZONE_CLEAR_CD to 0 and ALARM_TYPE_ID to "GREEN".

- MOPP_CON_TYPE – Contains identifiers, descriptions, and airbase map colors for each MOPP level.

Column	Type	Description
MOPP_CON_TYPE_ID	VARCHAR	MOPP level identifier
MOPP_CON_TYPE_LONG_NM	VARCHAR	MOPP level description
MOPP_CON_TYPE_COLOR	VARCHAR	MOPP level associated color

"MOPP 4" corresponds to a level of protective gear typically ordered when contamination is suspected. Since the ITAF provides a confidence level on zone contamination status, separate "MOPP 4" codes and descriptions are defined for various confidence ranges. These are listed below:

'10' – "MOPP 4"	Confidence unknown
'20' – "MOPP 4 - Conf. < 25%"	Confidence less than 0.25
'21' – "MOPP 4 - Conf. 25-50%"	Confidence between 0.25 and 0.5
'22' – "MOPP 4 - Conf. 50-75%"	Confidence between 0.5 and 0.75
'23' – "MOPP 4 - Conf. > 75%"	Confidence greater than 0.75

Process Flow

The mental model (primary use case) for ITAF application is as follows:

1. Prior to system use, sensors and other detection sources (with their initial reputations) are defined in tables CB_SITE and ITAF_SENSOR (sources include PSxx – portal shield, CSxx – chemical shield, and Mxx – detection paper).
2. Prior to system use, zones are defined in table ITAF_ZONE.
3. Prior to system use, ITAF_CONTROL parameters are set, including TIME_WINDOW_SIZE.
4. Prior to system use, associations between sensors and zones are defined in ITAF_SENSOR_ZONE.
5. Sensor detection readings are entered via the ITAF UI (manually or through upload). Readings are stored in table ITAF_SENSOR_READING_RECD. (Regardless of source, detection readings are referred to as "sensor readings" in all steps below.)
6. Through an ITAF UI, the user can invoke ITAF for a given time window. For each time window, ITAF performs the following:
    a. Using the window start time and the TIME_WINDOW_SIZE, ITAF gathers all included sensor readings in ITAF_SENSOR_READING_RECD and saves them in ITAF_SENSOR_READING_INPUT.
    b. ITAF determines the current sensor reputations by interrogating ITAF_SENSOR_REPUTATION for the reputations associated with the time window immediately preceding the time window being calculated. If no records exist, the initial reputations in ITAF_SENSOR are used.
    c. Using (i) the sensor-to-zone mappings in ITAF_SENSOR_ZONE, (ii) the sensor readings in the current time window (step a), and (iii) the current

sensor reputations (step b), ITAF determines zone assessments for all reported zones. The resulting assessments are saved in table ITAF_ZONE_STATUS_OUTPUT with corresponding confidence values.

d. ITAF updates the current sensor reputations in table ITAF_SENSOR_REPUTATION.

7. The user can view zone assessments in ITAF_ZONE_STATUS_OUTPUT using an associated ITAF UI. This output will also be used by the UCIRM for effects model analysis.
8. The user can view source reputations in ITAF_SENSOR_REPUTATION using an associated ITAF UI.

3.2.2.5 Determining Sensor to Zone Relationships

At Osan Air Force Base, sensors placed at various locations in the region of interest are used to detect chemical/biological contamination levels. The base is divided into multiple predefined NBC "zones," for the purpose of designating localized area of contamination and the ITAF determines these zone contamination states based on sensor inputs. To more accurately associate sensors with zones, UT:LIPS mapped each latitude/longitude point within a zone to the nearest sensor, instead of just associating the sensors with the zone(s) they are located in. The following algorithm was used to accomplish this mapping.

Given data
Zones were defined using selected latitude/longitude points along respective perimeters (18 zones). Sensor locations were also defined using latitude/longitude points (28 sensors).

Step 1. Zone Mapping
Assuming a latitude/longitude location grid to a resolution of 0.001 (degrees), all grid points in the region of interest were associated with a zone defined by the selected perimeter points (assuming longitude position value as [x], and latitude position value as [y]). Using the perimeter points, the perimeter of each zone was defined by piecewise linear lines, and inclusion of a given point in a zone was based on counting boundary contacts. When counting boundary contacts for a given x value, points inside the zone meet odd times with boundaries, but points of outside the zone meet even times with boundaries (Figure 13).

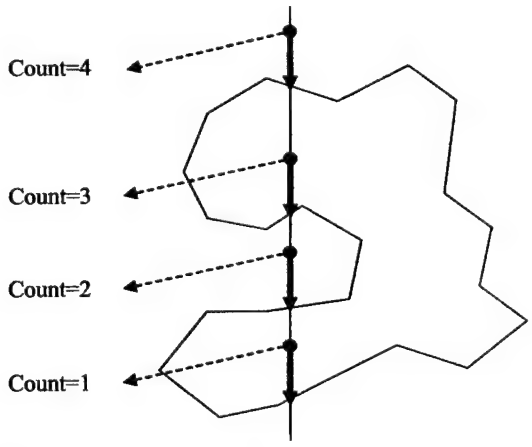

Figure 13: Location-to-Zone Mapping Using Border Examination

The flowchart in Figure 14 describes the zone mapping algorithm.

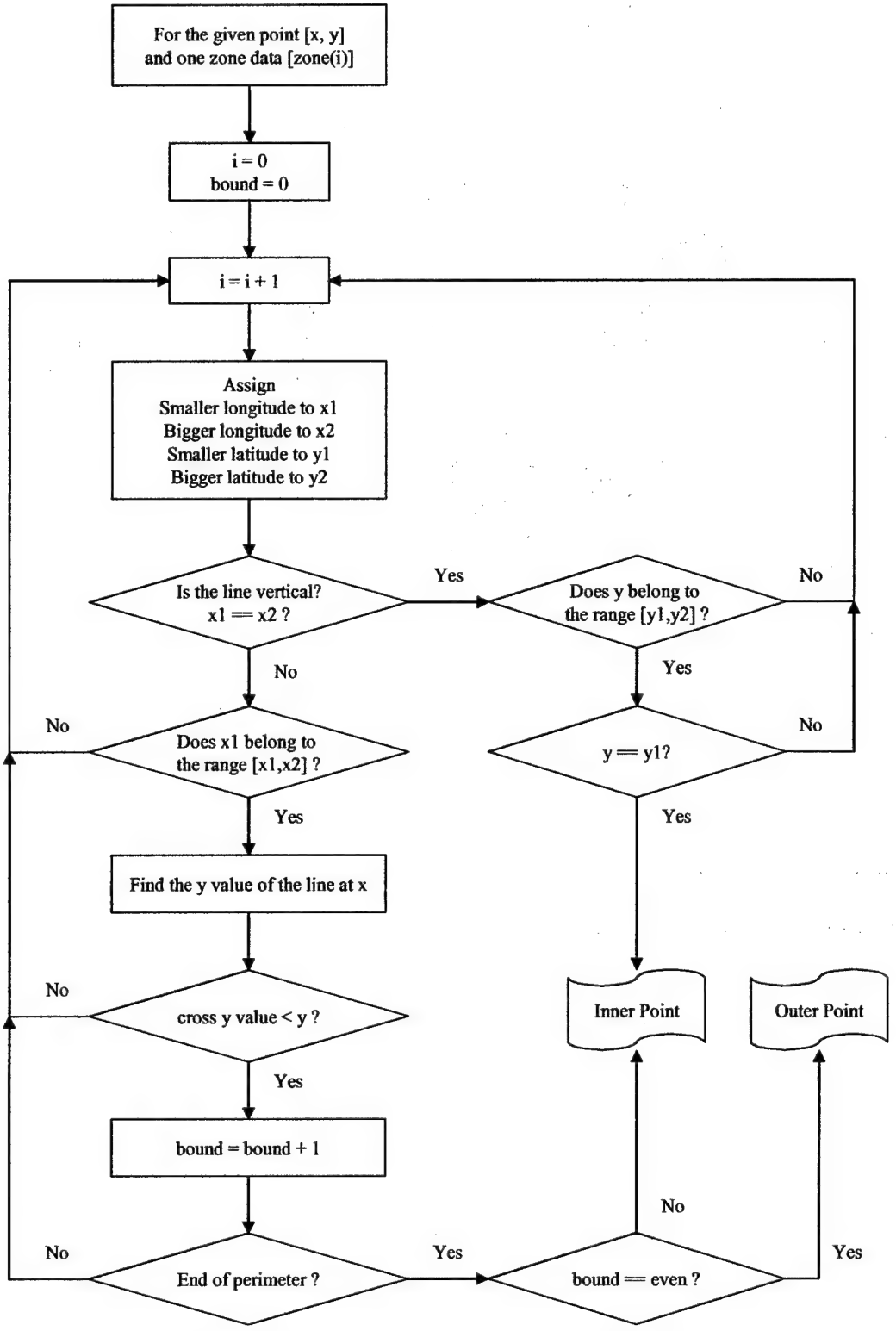

Figure 14: Location-to-Zone Mapping Algorithm

Figure 15 below shows the resulting location-to-zone assignments, where the blue asterisks represent given zone boundary points

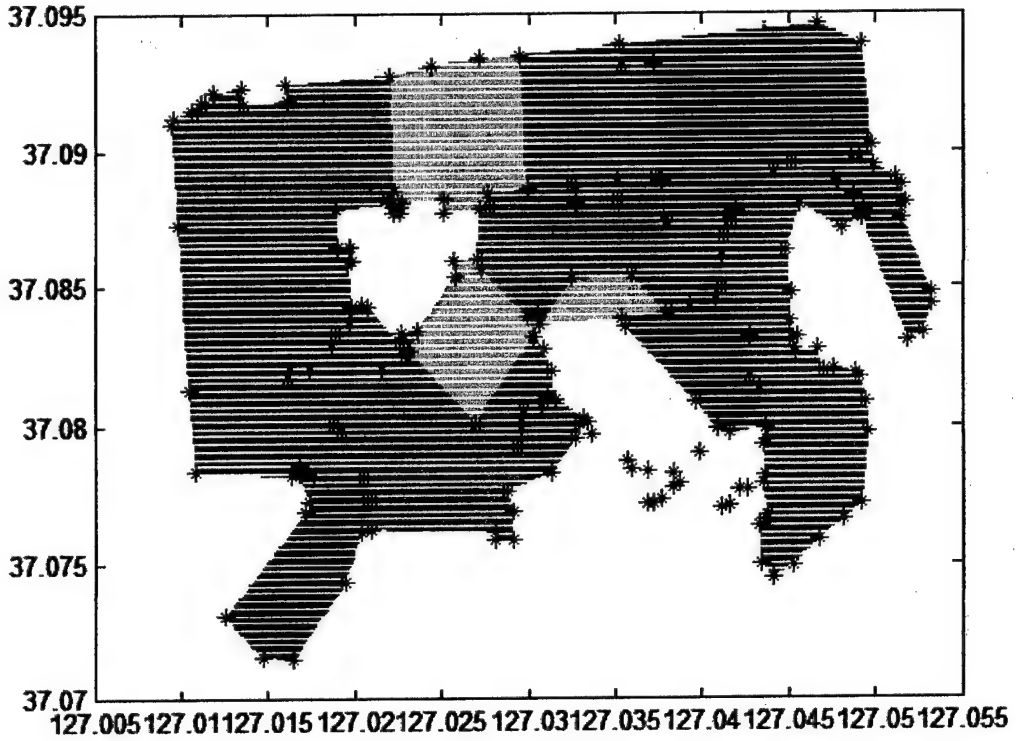

Figure 15: Location-to-Zone Mapping Results

Step 2. Sensor Mapping

Similar to zone mapping, each point in the grid must be mapped to the nearest defined sensor. Considering only locations inside the region of interest, the nearest sensor to each grid point was identified. Figure 16 depicts the results, where the blue asterisks represent defined sensors.

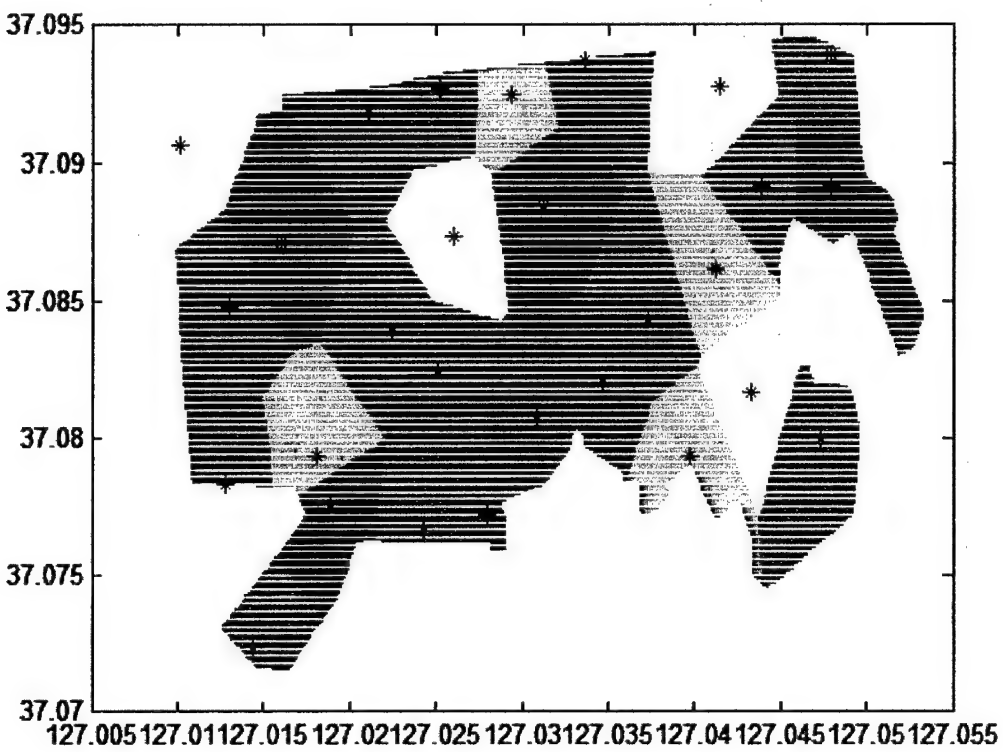

Figure 16: Location-to-Sensor Mapping Results

Step 3. Sensor-to-Zone Association

Results from Step 1 and Step 2 were combined, where each grid location was associated with a single sensor and a single zone, resulting in a mapping from sensors to zones. Then zones were associated with all sensors that had a nearest sensor relationship with any point in the zone. The results of this mapping compared to just associating the sensor with the zone(s) it is located in are shown in Table 2.

Table 2: Comparison of Zone Mapping Techniques

Sensor ID	Associated Zone IDs (w/o nearest-point mapping)	Associated Zone IDs (w/ nearest point mapping)
1	G, I	G, H, I, J, M, N
2	J	J, M, N
3	Q	Q, R
4	P, Q	J, P, Q
5	L, Q	L, Q
6	L, R	L, Q, R
7	L	L
8	L	K, L
9	A	A, B, D
10	A	A, C
11	E	C, E
12	F	F
13	G	G
14	E, G	C, E, F, G
15	C	A, C, E
16	C	A, C, D
17	D	C, D, E, G, H
18	K, M	B, K, M, N
19	M, N	J, K, L, M, N, O, P
20	B	B, K
21	O, Q	K, L, M, O, P, Q
22	D	B, D, H, M
23	G	G, I
24	F	F, G
25	B	B
26	K	K, L
27	J	J, P, Q
28	H	D, G, H, I

The significance of this enhanced resolution is that a much higher degree of confidence can now be obtained when a zone is assessed to have a "clear" status than if the sensor-to-zone associations were based only on the nominally "included" sensors. Given the potential lethality of chemical and biological agents, the desire to maximize confidence levels is obvious.

3.3. Incident Response Monitor / Effects Model

This section discusses topics regarding the Incident Response Monitor / Effects Model (IRM/EM) component of the UCIRM application. The following subsections address the underlying theory associated with estimating performance degradation due to a chem/bio incident and implementation of the IRM/EM component.

3.3.1 Theory Underlying the IRM / EM

3.3.1.1 UCIRM Solution Concept

The UCIRM approach to this problem is based on using geographical relationships to link incident domain data relevant to localized regions on an installation to mission-essential resources including the aircraft themselves.

Osan AFB has a known spatial relationship between an array of sensors that detect chemical and biological agents and a set of defined Nuclear, Biological, and Chemical (NBC) zones that cover the extent of the base. The intent is that the base will identify the particular level of required personal protection on a zone-by-zone basis. This defines the localized regions needed by the UCIRM solution concept.

In general, personal protection is inversely proportion to productivity; e.g., personnel in collective protection (i.e., a shelter) are not likely to be able to carry out the tasks required to support execution of the installation's missions. Use of personal protective equipment (PPE) for chemical and biological agents is commanded by Mission Oriented Protective Posture (MOPP) levels that range from MOPP 0 (least protected) to MOPP 4 (most protected). As the level of protection increases, the productivity of the personnel degrades. Several factors contribute to this degradation:

- Reduced vision due to the headgear
- Reduced tactile faculty due to the gloves of the PPE
- Required restriction of work/rest cycles due to exposure or thermal management

The UCIRM solution concept utilizes planning factors to quantify the level of degradation of the performance of mission essential tasks as a function of commanded protection on an NBC zone by NBC zone basis. Commanded collective protection will bring progress on mission-essential tasks to a halt. Commanded MOPP levels for individual protection result in specific degradation factors, represented by the MOPP level multi-factors in the model and a lengthened work rest cycle which compensates for the reduced ability of the body to dissipate heat while wearing protective gear.

Once the status of the required level of protection is known for the regions of the base, degradation factors can be applied to expected mission-essential task durations. ScenPro is developing an effects model that uses these relationships. A screenshot of that model is shown in Figure 17.

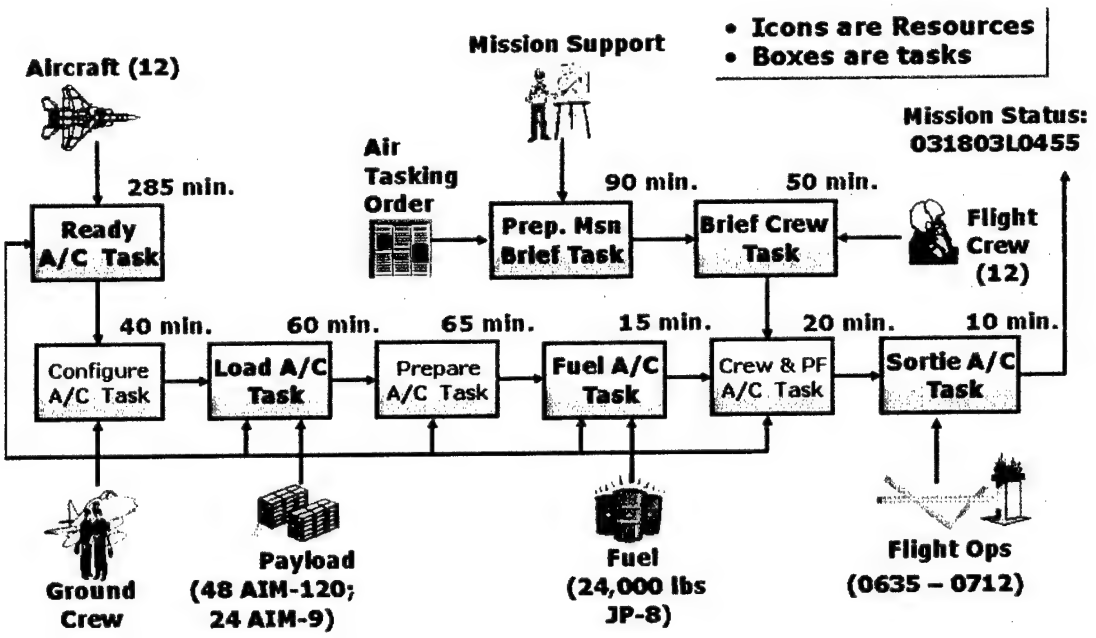

Figure 17: Effects Model Mission-Critical Resource Relationships

The model has a graphical representation of the critical resources and tasks that make up the process of preparing an aircraft for a mission. The model relates geographical locations to the following tasks:

- Ready Aircraft
- Prepare Aircraft
- Fuel Aircraft
- Load Aircraft
- Prepare Mission Brief
- Brief Flight Crew
- Crew and Pre-flight Aircraft
- Sortie Aircraft

The model includes elements for the following critical resources:

- Aircraft
- Ground Crew
- Fuel
- Payload (Ordnance or cargo)
- Flight Crew
- Mission Support
- Flight Operations

Relationships between geographic locations and critical resources are also used.

3.3.2 IRM / EM Implementation

3.3.2.2 Effects Model Implementation

ScenPro has created the effects model for the UCIRM application using a commercial modeling tool called Extend™. Extend is a product of Imagine That!, Inc. The application is well suited for task-based modeling of processes (such as preparing aircraft to fly missions as defined in an ATO). The application has the following features:

- The ability to model both discrete events and continuous timelines
- The ability include non-deterministic parametric values and stochastic processes in the model
- A library of pre-defined, user-configurable, functional blocks that model tasks related to material handling and transportation systems
- The ability to build user-defined functional blocks to supplement the ones available in the library
- An iconic representation of the modeled process with an animation function in the application's user interface, which helps the user understand the simulation process

Functional blocks in the process flow are represented by icons that define the individual steps in the process, as shown in Figure 18.

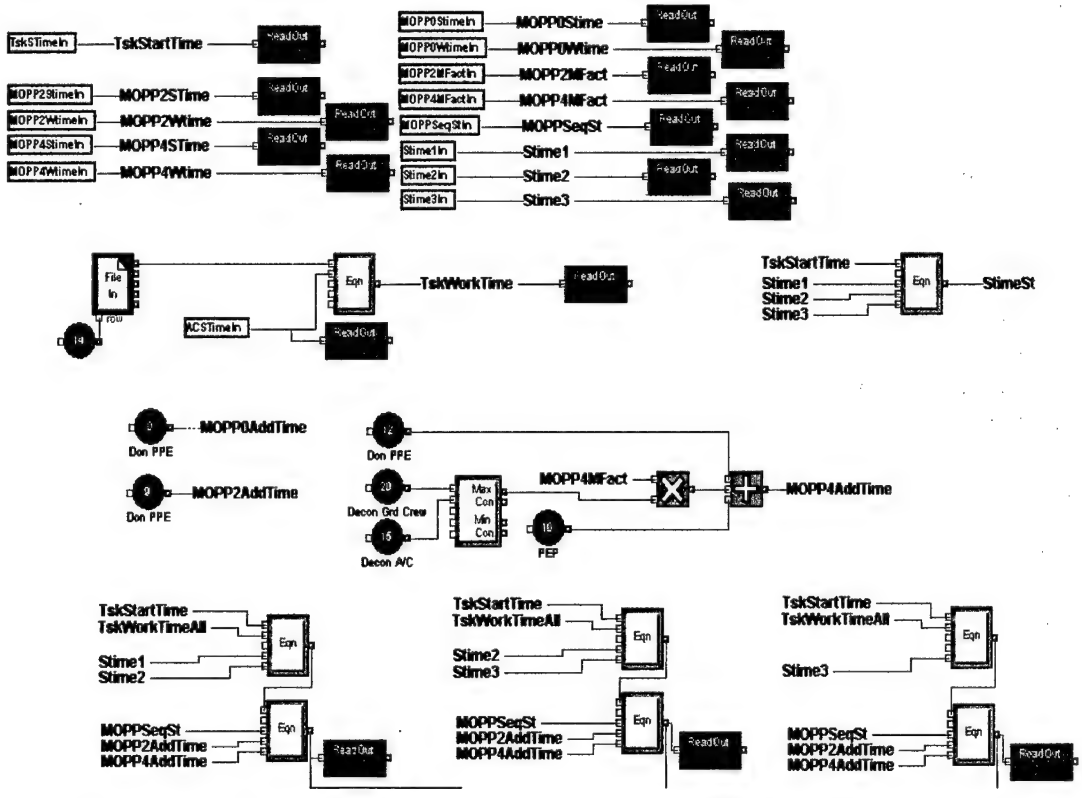

Figure 18: Effects Model Representation of Aircraft Readiness Task Flow

Figure 18 shows the steps in moving an F-16 aircraft from the state where it has returned from a mission to where it can be assigned to be prepared for a new mission. These model components represent a sequence of tasks that are performed on the aircraft icon on the left side of Figure 17. The top elements in the diagram read in the requirements for different levels of MOPP gear. The next layer of blocks calculates the amount of time the aircraft will take in terms of additional decontamination tasks and lengthening of task time due to MOPP gear reductions in efficiency. The rest of the boxes are used for mapping the transitions between MOPP level states for the application of work rest cycles. These tasks are the ones that would be affected by the degradation factors discussed in Section 3.3.1.1.

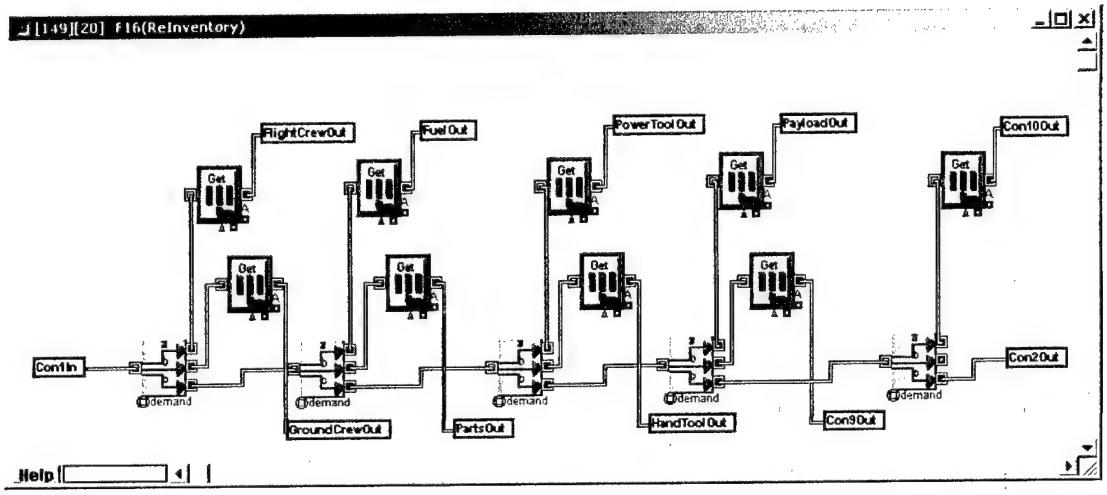

Figure 19: Effects Model Representation of F-16 Inventory Management

Figure 19 shows how resource quantities are handled in the model. As an aircraft is prepared for a mission, the model draws down an inventory for each resource type. The model treats some resources as renewable, in that once the task that this resource supports is completed, it becomes available again. An example of a renewable resource is a ground crew. Other resources, such as fuel, or payload, are non-renewable resources. The model elements shown in Figure 19 manage the task of determining the current quantities that are available for each resource.

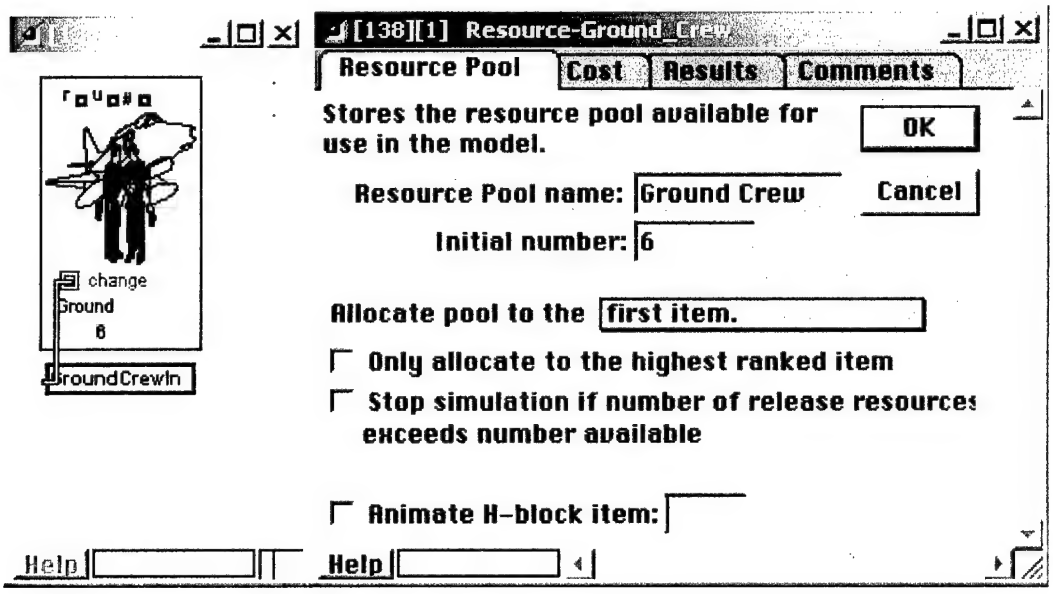

Figure 20: Effects Model Representation of Ground Crew Resource Interface

Figure 20 shows an example of how a resource in the model is defined and updated by the user. This example is for the Ground Crew icon shown in the lower left corner of Figure 17. Figure 20 shows the Extend™ user interface for a defined resource block. This interface allows the user to initialize the number of available resources and define a variety of other resource attributes.

4. Results and Discussion

4.1. ITAF Application Demonstration

The ITAF was designed to demonstrate the value of trust assessment in decision making by displaying sensor reputations and zone statuses over time. Screens were designed for data entry and to display two important considerations for military decision makers not available previously: (i) *ongoing sensor reputation* – allowing decision makers to emphasize sensors determined to be reliable and ignore sensors determined to be unreliable and (ii) *ongoing zone status and associated confidence* – allowing decision makers to determine the appropriate level of response to a potential incident based on level of confidence in detection of that incident. For this demonstration, the size of the "sliding time window" described in the prior section was fixed at one second, and time windows did not overlap.

Figure 21 depicts a screenshot of the ITAF. The figure shows four windows:

- "ITAF Test Data Generator and Control Panel" – Allows sensor readings to be entered or uploaded. Also provides a means for invoking the assessment facility for a given time window, generating zone assessment (clear or alarm) and confidence by considering clear/alarm assessments and confidences from relevant sensors and their respective reputations. The user can enter a threshold such that if sensor readings result in a zone "alarm" status with a confidence above the entered threshold, the zone MOPP level in the CZONE table is set to MOPP 4 with a description

corresponding to the appropriate confidence range. Each invocation of the assessment function results in an adjustment to the reputation for each sensor that provided data.

- "Zone Confidence Level Screen" – Displays current status for each zone, based on the last assessment performed. Also displays current MOPP level.

- "Sensor Reputation and Confidence Screen" – Displays current reputation for each sensor as of the last trustworthiness assessment invocation. Also displays the time, status, and confidence of the sensor's most recent reading.

- "ITAF Trend Viewer" – Allows the user to see the trend of zone statuses and sensor reputations over time. The Sensor Reputation graph shows the reputation between 0 and 1 for each sensor selected at each time window for which the assessment function was invoked. The Zone Status graph shows zone status, "alarm" or "clear," for selected zones at each time window for which the assessment function was invoked. Given that a status assessment is associated with a respective confidence, the graph displays status values using error bars proportional to degree of confidence.

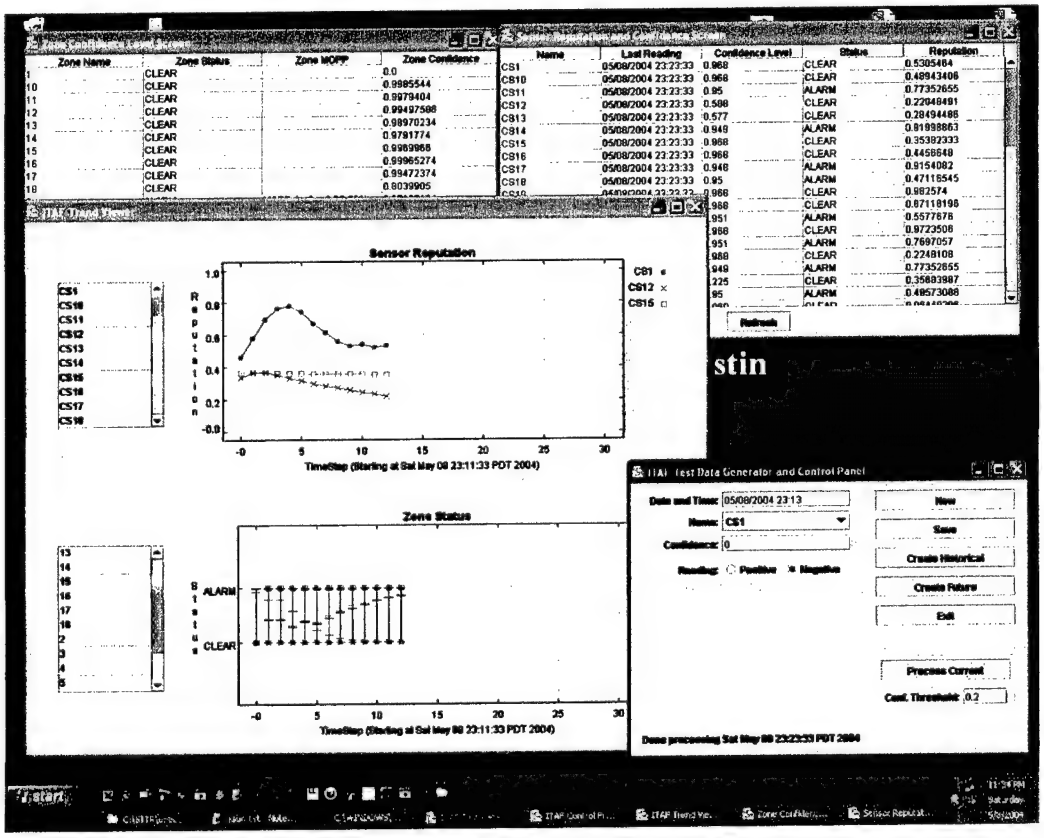

Figure 21: Screenshot of ITAF Application Windows

4.2. IRM / EM Demonstration

A Java program was created to store the Effects Model data in the AODB8 database in order to support the data needs of Extend while maintaining and sourcing data from that database. The program stores data required to run the Effects Model in the AODB8 database, extracts the data from the database in a format that the Extend model instance will use, and returns model results to the data base. For additional details on the screens shown in Figure 22 and their use, refer to the UCIRM: Effects Model User's Manual.

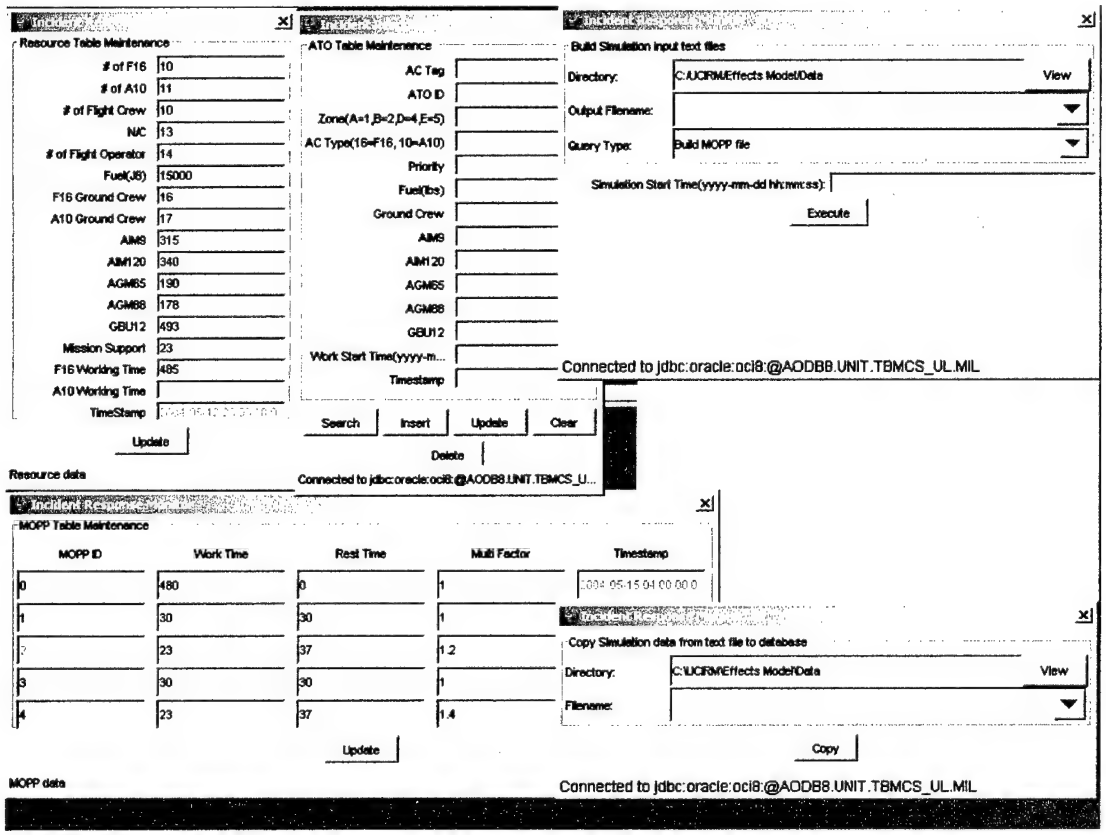

Figure 22. IRM - Effects Model Data Entry Screens

The following screens are provided for maintaining the data in the database:

Resource Table Maintenance: This screen allows the user enter inventory data for crews and munitions. It also allows the user to enter the notional times required for performing aircraft maintenance tasks without MOPP gear.

The fields and button in the screen are defined as follows:

Button

Update: This button records the data to the database when pressed.

Fields

Number of F-16: This field is the quantity of F-16s on the base.

Number of A-10: This field is the quantity of A-10s on the base.
Number of Flight Crews: This field is the quantity of pilots available to fly the airplanes.
N/C: This field is for future use.
Number of Flight Operators: This field is the number of control personnel available to manage flights.
Fuel: This field contains the number of pounds of jet fuel available at simulation start.
F-16 Ground Crews: This field represents the number of maintenance crew shifts available to ready an F-16 fighter plane. It is the key driver for completion time.
A-10 Ground Crews: This is the number of maintenance crew shifts available to ready the A-10 tactical plane. It is the key driver for completing an A-10.
AIM-9: This field contains the number of AIM-9 missiles available at the beginning of the simulation. The ATO will assign a specific number to a planned flight.
AIM-120: This field contains the number of AIM-120 missiles available at the beginning of the simulation. The ATO will assign a specific number to a planned flight. This is the AMRAAM missile.
AGM-65: This field contains the number of AGM-65 missiles available at the beginning of the simulation. The ATO will assign a specific number to a planned flight.
AGM-88: This field contains the number of AGM-88 missiles available at the beginning of the simulation. The ATO will assign a specific number to a planned flight. This is the High-speed Anti Radiation Missile.
GBU-12: This field contains the number of GBU-12 bombs available at the beginning of the simulation.
F-16 Working Time: This field contains the number of minutes that it takes to prepare an F-16 for flight.
A-10 Working Time: This field contains the number of minutes that it takes to prepare an A-10 for flight.
Mission Support: This field is the number of teams available to support operation planning for flights.
Timestamp: This field contains the last time the records were modified.

ATO Table Maintenance: This screen allows the user to enter data about scheduled aircraft flights including payload, work start time, priority and location. The data entered through this screen sets up the aircraft list that will run through the extend model with the start time being an offset from simulation start time. The location entered here is mapped by Extend into the zones produced by the MOPP zone query of the Build Simulation data Screen. The ATO ID and Aircraft Tag numbers should be unique since this is used to track the results stored from Extend.

The following fields and buttons are from the ATO Table Maintenance Screen.

Button

Search: After entering in the tail number of a plane in the database, the search button will bring up the aircraft to be modified for its new mission.
Insert: This button will save a new aircraft after the fields have been entered.

Update: After an existing aircraft has been modified, the record is saved to the database by pressing the update button.
Clear: This button will clear all the field entries on the screen without saving.
Delete: This button removes the aircraft from the database.

Fields

Aircraft (AC) Tag: This field contains the tail number of the aircraft that a flight operation plan is being created for. It is in the form of a number such as 100010.
ATO ID: This field is a unique identifier to identify the AC on the ATO. The data in this field should not be duplicated for any other aircraft.
Zone: This field identifies the base chemical zone that the work on the plane is going to occur in. Rome Lab's digital dashboard uses letter codes on the common operational picture (COP) map and Extend does not allow for letters in input fields. The data entered into this field is in numeric format and should be converted from a letter to a number (A=1, B=2, etc.).
Aircraft (AC) Type: This field indicates if the flight will be performed by an F-16 or an A-10. "10" represents an A-10 and "16" represents an F-16 in the Extend model.
Priority: This field contains the sequence that the plane receives critical resources.
Fuel: This field contains the quantity of fuel that the plane will need for a mission. This is compared to the available fuel consumed by the proceeding aircraft based
Ground Crew: This field is the number of ground crews that will be assigned to the aircraft, if available when they are drawn from the pool. The model will take up to this many crews from the pool. If no crews are available the aircraft will wait until additional crew(s) are available.
AIM-9: This field represents the quantity of AIM-9 missiles that the plane will carry on its mission.
AIM-120: This field represents the quantity of AIM-120 missiles that the plane will carry on its mission.
AGM-65: This field represents the quantity of AGM-65 air to ground missiles that the plane will carry on its mission.
AGM-88: This field represents the quantity of AGM-88 air to ground missiles that the plane will carry on its mission.
GBU-12: This field represents the quantity of GBU-12 bombs that the plane will carry on its mission.
Work Start Time: This field is the time that the ground crew will begin preparing the aircraft for its mission. It is converted to minutes relative to the start time that is entered in the Build Simulation Input Text Files. In general, this time and date will be calculated from the estimated time of departure (ETD) for the aircraft.
Timestamp: This field is display only and contains the date and time the record was last modified.

Build Simulation Input Text File: This screen provides the user with the ability to create the three Extend input files, ATO.txt, Resource.txt, and ZoneMOPP.txt through the use of three queries to the database. This screen is used by selecting the file names in presented from the directory field and loading them with the pre-built queries that are selected from the 3rd field. The simulation start-time that all the Extend displays will use as the 0 time is also entered through this screen.

The following fields and buttons are from the screen displayed in Build Simulation Input Text File.

Buttons

View: This button creates a list of all the files identified in the Directory field and places this list in the Output Filename dropdown.
Execute: This button executes the stored query identified in Query Type and its data placed in the file identified in Output Filename.

Fields

Directory: This field contains the directory where the data will be stored. Extend uses absolute addresses and the values in this field should not be changed from the default.
Output Filename: This drop-down field allows the user to choose a filename from those files in the previously defined directory. The file name should match the query according the following table.

Filename	**Associated Query**
ATO.txt	Build ATO File
Resource.txt	Build Resource File
ZoneMopp.txt	Build MOPP Zone File

Query Type: This dropdown field contains the database requests for creating the input file. See the table above for a mapping of queries to correct filenames.
Simulation Start Time: This field is used as the simulation start time and all calculations are derived from this number. The data must be in the format: 'yyyy-mm-dd hh:mm:ss'.
Simulation End Time: This field is used to determine the dates and times used out of SCPR_CZONE_HISTORY for creating the MOPPZone.txt file.

MOPP Table Maintenance Screen: This screen allows the user to enter data about the work-rest cycles and the impact to aircraft maintenance task times due to MOPP gear through the multi-factor. The multi-factor is a number that stretches the time needed to perform a task due to the reduction in visual acuity and tactile function of the MOPP gear.

The fields in the MOPP Maintenance screen are as follows:

Button

Update: This button writes the changes made on the screen to the database.

Fields

MOPPID: This field is not modifiable and represents the 5 MOPP levels.
Work Time: This field allows the user to identify the number of minutes that a person can operate at the MOPP level.

Rest Time: This field allows the user to identify the number of minutes that the person must rest after completing a work cycle.
Multi-Factor: This field allows the user to enter the number used to extend the amount of effort due to MOPP gear reducing visual acuity and tactile function.
TimeStamp: This is a display only field indicating when the record was last modified.

Copy Simulation Data Screen: This screen allows the user to place the results, which are stored in the SimResults.txt file, of the simulation back into the database for future use.

The fields and buttons used on the Copy Simulation Data screen are described below:

Buttons

View: This button creates a list of all the files identified in the Directory field and places this list in the Filename dropdown.
Copy: This button executes the stored insert commands and moves the data in the Extend output file into the database.

Fields

Directory: This field contains the directory where the data will come from. Extend will always use same directory location and values in this field should not be changed from the default. Both input files and output files are in the same directory.
Filename: This drop-down field allows the user to choose filename from those files in the directory. The file name chosen should be "SimResult.txt".

The Build Simulation Input Text File and the Copy Simulation Data screens are used to transfer data to and from the Extend Effects Model, as described in section 4.2.1, Effects Model Architecture.

The user will use the data entry screens to set up the planned flights for the day based on an ATO, determine the resources available for the simulation, and use the data for attacks stored in the database to explore the impact of attacks and what the application of different crew levels and priorities will do to the timeline for meeting the mission goals.

4.2.1 Output

Three output displays are provided for the user from the Extend Model in addition to the SimResults.txt and SCPR_SIM_RESULT table. The first output is the current run showing the amount of time it took to complete the aircraft and is described in section 4.2.1.1. The other two outputs show the last runs of the simulation so a side by side comparison can be made of the different runs. These are described in section 4.2.1.2.

4.2.1.1 Current Plot of Data

The [10][0] Discrete Plotter shown in Figure 23 shows a plot of quantity of planes completed by number of minutes from simulation start time. This is the basic output of the program and will show different results for each scenario.

The plot's x-axis shows the number of minutes from simulation start time that the airplane was completed. The y-axis indicates the cumulative quantity of aircraft completed at a given time. The simulation ends at the user defined maximum time which defaults to 10,000 minutes.

The table describes the plot and allows the user to get detailed information about the specific time the aircraft is completed. The first column, Point Number, is an index representing the number of readings recorded on the plot. The second column, Time, is the times the F16s are completed it will always have a 0 indicating start and 10,000 representing the final time of the simulation. The third column, F-16 Ready, shows the values of the y-axis of the plotter and is the cumulative number of F-16 aircraft completed at the time listed in the previous column. The third column, Time, is the time that an A-10 is completed. The 4th column of interest is A-10 Ready. This column is the cumulative number of A-10 aircraft completed.

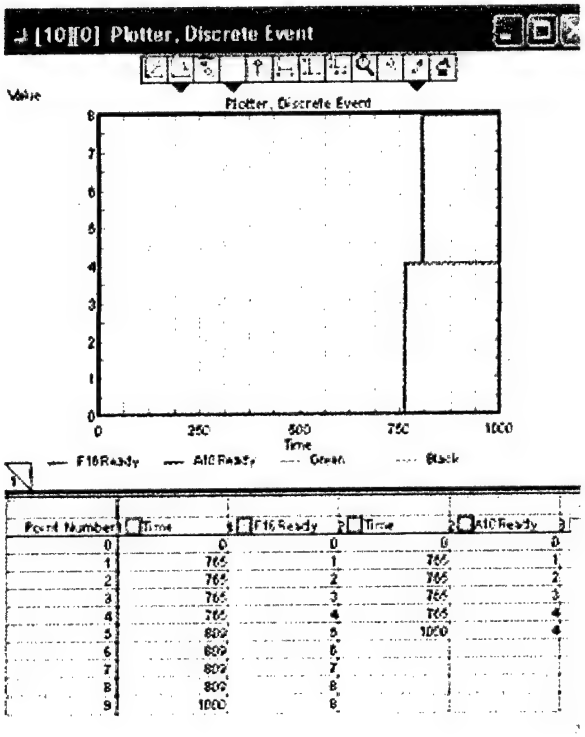

Figure 23. Current Extend Run Output

4.2.1.2 Multi-Run Plots of Data

The Multi-Run Plotters shown in Figure 24 shows a plot of the last runs performed by the Extend model for F-16s and A-10s. These plotters will allow the user to compare different COAs involving additional crews or different aircraft sequences.

These plotters retrieve the last runs performed during the session. The plot is the same as the current run, x-axis is minutes from simulation start time and the y-axis is the cumulative number of aircraft completed..

The table has alternating rows of older runs after the Point Number column. The 2nd, 4th, etc. columns containing the completion times in minutes relative to simulation start for the aircraft. The third, 5th, etc. columns contain the cumulative aircraft completed for each previous run.

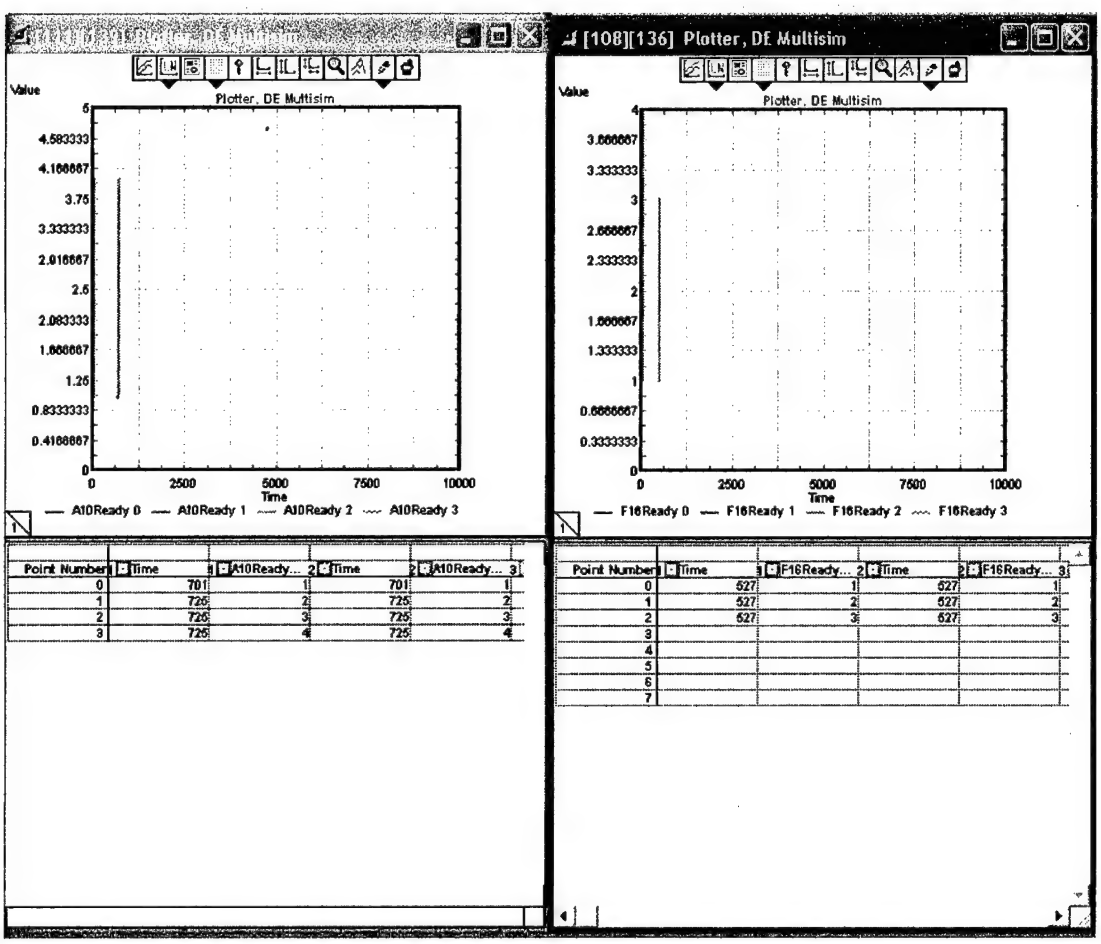

Figure 24. F-16 and A-10 Multi-Run Plots

4.2.2 Effects Model Architecture

Figure 25 shows the data transfers and storage used by the IRM application to implement the effects model

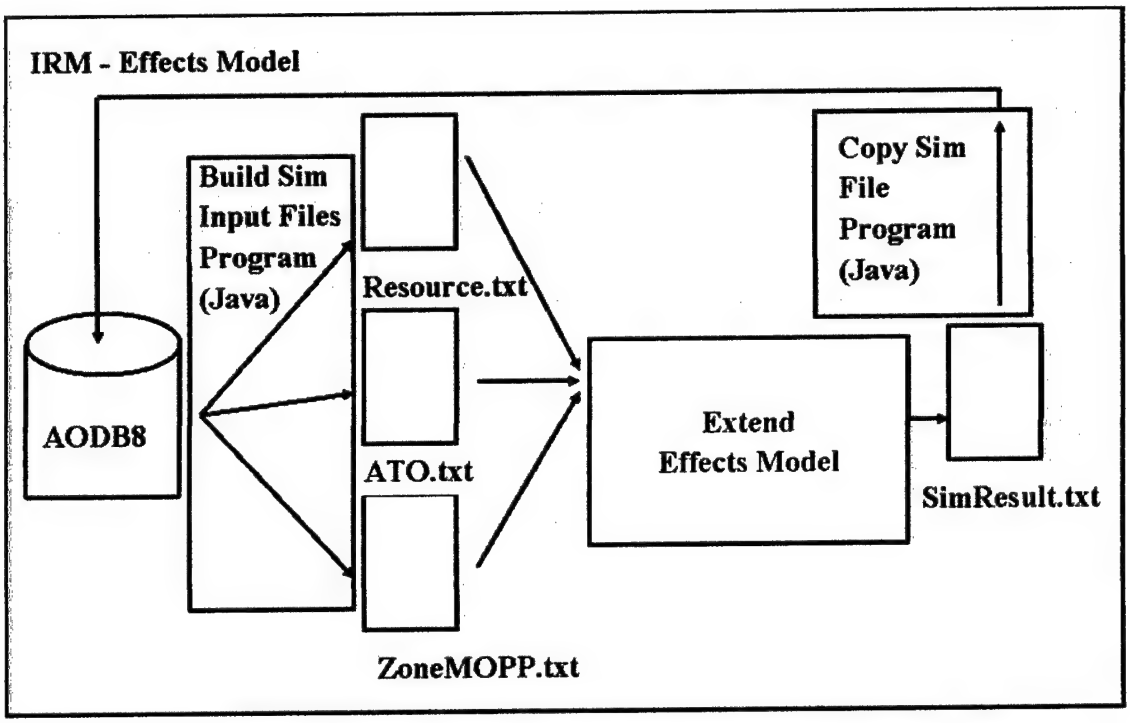

Figure 25. IRM - Effects Model Architecture

4.2.3 Database Tables

The following tables are used by the effects model. In a production environment, these tables would receive the data from a variety of information sources around the base. For example, the sensor data results would be collected in the CZone table then transferred to the SCPR_CZONE_HISTORY table, while the resource table would receive feeds from the on-base inventory systems.

- SCPR_ATO_DETAIL
- SCPR_CZONE_HISTORY
- SCPR_MOPP_MSTR
- SCPR_RESOURCE
- SCPR_SIM_RESULTS

A detailed description of these tables is follows.

- SCPR_ATO_DETAIL – Contains information about scheduled flights, their preparation start times, and payloads.

Column	Type	Description
ATO_ID	Number	Unique identifier of a line in a ATO.
ZONE	Number	Numeric equivalent of the alphabetic chemical zones of the base
AC_TAG	Number	Unique identifier of the aircraft that represents its tail number
AC_TYPE	Number	The aircraft type expressed as a number: 16 = F-16 10 = A-10
PRIORITY	Number	This is the order that aircraft receive fuel, munitions, and crews.
Fuel	Number	This is the amount of fuel an aircraft will be loaded with for the mission.
Ground Crew	Number	This is the number of crews that are dedicated to the aircraft.
AIM9	Number	This is the number of AIM-9 missiles to be loaded on the aircraft.
AIM120	Number	This is the number of AIM-120 missiles to be loaded on the aircraft.
AGM65	Number	This is the number of AGM-65 missiles to be loaded on the aircraft.
AGM88	Number	This is the number of AGM-88 missiles to be loaded on the aircraft for the mission.
GBU12	Number	This is the number of GBU-12 bombs to be loaded on the aircraft for the mission.
WORK_START_TIME	Date	This is the date that work will begin on the aircraft.
FUTURE_USE1	Number	Reserved for use in future applications
FUTURE_USE2	Number	Reserved for use in future applications
DTTM	Date	This is the time and date when the record was last modified

- SCPR_CZONE_HISTORY – This table represents a historical record of the MOPP and alarm levels for each of the bases chemical zones.

Column	Type	Description
LOG_ID	Number	This is a unique identifier for the record.
CZONE_ID	VARCHAR	This identifies the chemical zone that this record addresses.
CZONE_NAME	VARCHAR	This is the name of the chemical zone.
MOPP_STATUS	Number	This number represents the what MOPP level the zone was at when this record was created. The information is sourced from MOPP_CON_TYPE. MOPP0 = 1 MOPP1 = 2 MOPP2 = 3 MOPP3 = 9 MOPP4 = 10,20,21,22,23
ALARM_STATUS	Number	This indicates if the zone was CLEAR or ALARM.
LOG_DTTM	Date	This is the date and time the zone was at the recorded MOPP and Alarm level.
DTTM	Date	This is the date and time the record was last modified.

- SCPR_MOPP_MSTR – This table contains the information about the work rest cycles and the factor for extending work times due to MOPP levels.

Column	Type	Description
MOPP_SEQ_NUM	Number	This is a unique identifier for the record.
MOPP_ID	Number	This number is the MOPP level 0-4.
WORK_TIME	Number	This is the maximum number of minutes a person can work at the MOPP level.
REST_TIME	Number	This is the minimum number of minutes a person must rest after working at the MOPP level.
MULTI_FACTOR	Number	This factor represents the additional amount of time the task will take due to reduction of visual acuity and tactile function due to the MOPP gear.
DTTM	Date	This is the date and time the record was last modified.

- SCPR_RESOURCE – This table contains the information about inventory of munitions, fuel and human resources. This table also includes the times for completing an aircraft under MOPP0 conditions.

Column	Type	Description
RES_SEQNUM	Number	This is a unique identifier for the record.
RES_TYPE	VARCHAR	This is the name of the resource to be stored.
RES_DESC	VARCHAR	This is a description of the resource.
UNIT	VARCHAR	This is the unit that the resource quantity is expressed in.
Quantity	Number	This is the amount of the resource on hand at the start of the simulation.
DTTM	Date	This is the date and time the record was last modified.

- SCPR_SIM_RESULTS – This table contains the information from after the simulation is run and describes the number of minutes that the aircraft will be completed relative to the simulation start time.

Column	Type	Description
ATO_ID	Number	This is the ATO_ID used for the aircraft from the SCPR_ATO_ID.
AC_TAG	Number	This uniquely identifies the aircraft that as used in the simulation.
AC_TYPE	Number	This is the AC type as a number. 16 = F-16 10 = A-10
AC_START_TIME	Number	This is the time that AC started to be prepared for a mission expressed as minutes from the simulation start time.
AC_END_TIME	Number	This is the time the AC was completed and had become airborne.
DTTM	Date	This is the date and time the record was last modified.

5. Conclusions

The User Configurable Incident Response (UCIRM), designed and implemented jointly by ScenPro, Inc. and the Laboratory for Intelligent Processes and Systems at the University of

Texas at Austin (UT:LIPS), provides decision support for military personnel responsible for monitoring military installations to determine if designated zones (i.e. predefined regions) have been subject to chemical/biological attack. Contamination assessment is rendered based on detection by multiple fixed automated sensors as well as manual sampling. Specifically, the UCIRM supports personnel with respect to the following two decision-making tasks:

- *Estimating how zone contamination will affect the tempo of base operations:* The consequence of marking a zone contaminated is that ground crew working in that zone must wear protective clothing (i.e., MOPP gear) until the contamination dissipates. Ground crew cannot operate at maximum performance while wearing this gear, thereby increasing the time required to service aircraft and negatively impacting the current mission schedule. The UCIRM's Effects Model (EM) component estimates how zone contamination will affect a given mission schedule, determining when designated aircraft as a function of task models, available resources, and crew productivity, qualified by predicted performance degradations resulting from protective gear.

- *Determining zone contamination with an assigned confidence level based on the trustworthiness of sensor readings and the sensors providing those readings:* The UCIRM's Information Trust Assessment Facility (ITAF) helps monitoring personnel assign a zone contamination status with a respective confidence level supported by multiple information sources, including sensor readings and manual samples. Assessment considers (i) degree of corroboration among information sources; (ii) confidence levels assigned by those sources for each clear/alarm reading; and (iii) source reputations, which are maintained over time. In particular, this approach helps avoid potential knee-jerk reaction to false alarms, where a zone is marked as contaminated based on an alarm raised by a faulty sensor, thereby unnecessarily degrading performance in that zone.

The UCIRM was designed to integrate with the Digital Dashboard application developed by AFRL Rome Labs. All UCIRM data is stored in the Air Operation Database (AODB) used by the Digital Dashboard, and the Dashboard's mapping facility displays MOPP level for each zone, which is set by the UCIRM when contamination is determined.

As part of a Small Business Technology Transfer (STTR) effort, this contract resulted in two primary deliverables:

- *Demonstration:* The UCIRM can be demonstrated using a chemical/biological attack scenario at a major air base devised by ScenPro and UT:LIPS. The demonstration includes both ITAF and EM functionality, and can be run standalone or in conjunction with the Digital Dashboard.
- *Dissemination:* Research advances in trustworthiness assessment technology and experiences regarding the application of that technology were disseminated through publications, briefings, and technical exchange at conferences.

AFRL has shown considerable interest in the UCIRM, particularly given the additional functionality the UCIRM provides the Digital Dashboard. Program participants are investigating possible follow-on work to enhance the UCIRM and further integrate it with the Digital Dashboard.

6. Dissemination of Program Effort

Publications

UT:LIPS has produced the following publications documenting research performed under this contract.

- K. Suzanne Barber, Jisun Park, "Robust Partner Selection Scheme for Information Quality Assurance despite Uncertainty in Open Multi-Agent Systems," In Proc. of IEEE International Conference on Information Technology (ITCC04), Vol. 1, pp.430-434, Las Vegas, NV, April, 2004.

- K. Suzanne Barber, Jisun Park, "Finding Partners to Form Information Sharing Networks in Open Multi-Agent Systems," (to be published) In Proc. of the 17th Florida Artificial Intelligence Research Symposium (FLAIRS04), Miami, FL, May, 2004.

- Jisun Park, K. Suzanne Barber, "Finding Information Sources by Model Sharing in Open Multi-Agent Systems," submitted to the Workshop on Agents for Ubiquitous Computing (UbiAgents) at AAMAS 04, July, 2004.

- K. Suzanne Barber, Jisun Park, "Agent Belief Autonomy in Open Multi-Agent Systems," submitted to Computational Autonomy, Lecture Notes in Computer Science.

- Jisun Park, K. Suzanne Barber, "Model Sharing for Agent Belief Autonomy in Open Multi-Agent Systems," Technical Report, TR2004-UT-LIPS-002, The University of Texas at Austin.

As research progresses in the area of information trustworthiness assessment in decision support, UT:LIPS will continue to publish new findings, acknowledging this contract as appropriate.

Conferences

In addition to disseminating research advances through publications, UT:LIPS participated in academic conferences to conduct technical exchange with others in the Artificial Intelligence research community. Specifically, UT:LIPS attended and participated in the *18th International Joint Conference on Artificial Intelligence (IJCAI-03)* and the *2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2003)*.

Meetings

ScenPro and UT:LIPS were invited to present a briefing on the UCIRM development effort to a panel arranged by the Joint Operational Effects Federation (JOEF) project. Members of this panel included the JOEF project manager, the JOEF Systems Engineering Integrated Product Team lead, and representatives from the Army, the Air Force, and the Defense Threat Reduction Agency (DTRA). The stated goal of the panel was to assess the maturity and

potential near-term utilization of software applications related to managing the threat posed by chemical and biological weapons. Mr. Swenholt and Dr. Graser attended the panel on March 26, 2003 and jointly presented the briefing.

6.1. Interests

As described in prior sections, the UCIRM was designed to interface with the AFRL/Rome Digital Dashboard application through the AODB, enhancing existing Dashboard functionality. As such, AFRL/Rome management has expressed interest in further enhancing and promoting the UCIRM and increasing the level of integration between UCIRM and Dashboard operation.

6.2. Transitions

ScenPro will investigate whether the CASPOD ACTD can arrange to help fund a demonstration of the prototype UCIRM as part of the ACTD effort. We have discussed this with Mr. James Reilly of AFRL/IFSF, who is a member of the Information Technology Working Group for the CASPOD ACTD.

6.3. Consultative and Advisory Functions

As mentioned above, ScenPro believes that there is a valuable opportunity to work with the Information Technology Working Group (ITWG) of the CASPOD ACTD. Our point of contact with the ITWG will be Mr. James Reilly of AFRL/IFSF, who was the technical representative for the government during the Phase I effort of this STTR program.

Mr. Reilly was on the ITWG of the RestOPs ACTD and was involved in the development of the Digital Dashboard application that we used as a display mechanism for the UCIRM application. Mr. Reilly has worked with both ScenPro and UT:LIPS in the past on other efforts. He has asked that ScenPro consider and advise his organization how this effort might support the CASPOD ACTD.

6.4. Background

This section provides background on ScenPro, Inc. and the Laboratory for Intelligent Processes and Systems with regard to their experience, resources, and competencies in the domain of decision support for military personnel and trustworthiness assessment research.

6.4.1 ScenPro

ScenPro has been involved in programs that provide software applications and systems support to the military planning and incident response communities for years. Our products include information management and decision support software applications that were developed for a variety of customers in the Army, Navy, and Air Force. The primary function of many of these products is the collection and display of the information needed and used by incident responders or other users of mission-critical information. A description of several of these programs follows.

6.4.1.3 Chemical/Biological Incident Response Tool

The Chemical/Biological Incident Response Tool (CBIRT) was developed during a Phase II SBIR program for the Air Force Research Laboratory at Rome, NY. The tool enhances an

Incident Commander's ability to effectively plan for, manage, and mitigate the consequences of an incident that involves chemical warfare agents, biological warfare agents, or hazardous materials by providing the following capabilities:

- Capture key attributes of the incident and the associated response effort over time
- Provide map-based, visual display of key incident response elements
- Show quantitative assessment sufficient to answer the question "How's the response effort going?"Provide on-demand drilldown from top-level status to detailed task data
- Support for real-time tracking of response execution and the likelihood of success assessment for a pre-selected Course of Action
- Identify medical treatment requirements and determine discrepancies between available and required medical treatment resources
- Playback recorded incident and response 'state' data to support training and after-action reviews

Key features: collection and management of WMD-related incident response information.

6.4.1.4 Agent Supported Information Visualization

The Agent Supported Information Visualization (ASIV) program is a Phase II SBIR currently in work for the Air Force Research Laboratory at Rome, NY. ASIV provides intelligent push of mission specific information to small unit soldiers during planning and mission execution. Mission and role schemas residing in ASIV identify information the user may need during planning and execution. The agent infrastructure uses the schema to identify data sources and tailor system responses to user needs. Data characteristics include analyzing dynamic message traffic, maps, database information, and imagery.

The information needed to plan and execute a small unit mission comes from many sources. A publish-and-subscribe mesh/grid provides a means of extracting relevant content from multiple representations and fusing information from data sources to create new information, customized to the situation. ASIV brings data feeds together, and controls what is sent (status reports, deltas) and when (scheduled, when a change occurs), when requested to individual soldiers and team leaders, as a function of their information management plan and user profiles.

Throughout its development, ASIV uses scenarios and domain experts to identify the information needs, information availability, display preferences, and situation constraints in the flow of information to and from the user. Our ontology development, which supports intelligent filtering, recognizes mission and roles information needs and provides a plug-and-play capability to add new mission or self select portions of the ontology relevant to each new mission.

Key features: software agent technology; relevant information extraction for mission planning

6.4.1.5 Biological/Chemical Incident Response Monitor

ScenPro submitted a winning proposal to the Air Force Research Laboratory at Rome, NY for a Phase I STTR program in 1999. The system concept that was described in that proposal was for a Biological/Chemical Incident Response Monitor (BCIRM) that focused on selecting

and presenting information in the Incident Response domain. Between the contract award and the kickoff meeting, the customer directed that the focus be from the perspective of Air Force Command and Control (C^2). During the execution of the program, we identified the key issue of the assessment of the affects of the incident in the language of the Operational domain. The system architecture development focused on that aspect from that point forward, and we came up with the concept of using an effects model to realize a solution for the problem of extracting meaningful operational domain information from incident response domain data.

Prior to the generation of the Phase II proposal, AFRL asked that the operation of the tool be generalized to the point where it could operate with diverse user roles in scenarios outside just Chem/Bio and Air Force C^2. In response, we generalized the conceptual operation of the proposed tool, and changed the name from Biological/Chemical Incident Response Monitor to User Configurable Incident Response Monitor (UCIRM) to reflect the more generalized operation. The proposal used as the basis for this contract award was written from that perspective.

Key features: Translation of information via effects from the Incident Response to the Operational domains.

6.4.2 Laboratory for Intelligent Processes and Systems (UT:LIPS)

The Laboratory for Intelligent Processes and Systems at the University of Texas at Austin (UT:LIPS) conducts research in two focus areas currently receiving critical attention: formal system engineering process methodologies and distributed agent-based systems, the latter being closely related to the UCIRM effort. Specifically, the UT:LIPS "Sensible Agents" program involves the practical deployment of distributed multi-agent systems (MAS) composed of autonomous software agents designed to aid decision making in a designated problem domain. "Sensible Agents" offer a number of advanced capabilities to augment human decision-making in complex and dynamic environments. Specific capabilities include the following:

1. *Trust Evaluation:* assessing the reliability of information and information sources;
2. *Dynamic Organization Formation:* selecting the best decision-making frameworks given a set of goals and the state of the situation (dynamically deciding who should help plan for a goal and who should be bound to execute those plans); and
3. *Collaborative Planning and Execution:* coordinating planning and execution among distributed decision-makers.

To facilitate empirical analysis in the above research areas, UT:LIPS has developed the Sensible Agent (SA) Testbed, which offers an environment for distributed agents to operate and communicate and provides an easily configured framework for conducting repeatable experiments. Using an industry standard communication infrastructure (CORBA), the SA Testbed operates in a distributed, heterogeneous computing environment where agents residing on multiple platforms communicate with each other and third party external environments (e.g. simulation environments). Formal intra- and inter- SA interface specifications and Testbed utilities permit rapid set-up for experiments; rapid infusion of new advancements (e.g. algorithms, model representations, communication/collaboration mechanisms) as the research progresses; plug-and-play of third party technology

contributions for comparison studies; and connection with both remote users and remote simulation environments.

Organizationally, UT:LIPS resides in the Electrical and Computer Engineering Department and physically resides within the Applied Computational Engineering and Sciences building (ACES). Opened in August 2000, ACES was designed for interdisciplinary research and graduate study bringing together faculty, graduate students and research resources from three closely related programs at the University of Texas: Electrical and Computer Engineering, Computer Sciences, and the Texas Institute of Computational and Applied Mathematics, focusing on the state of the art in software engineering, visualization and graphics, intelligent systems, and parallel and distributed systems. This interdisciplinary research community provides a fertile environment for graduate education: training the next generation of scholars, scientists, and engineers. ACES boasts 16,500 square feet of lab space and high-speed networks. UT:LIPS facilities include office and lab space for 15 students and 6 professional research staff and visitors (approx. 3000 square feet). Demonstration of the proposed research results will be promoted by ACES' 14 seminar rooms, Auditorium, and 360 degree Visualization Lab immersing users. UT:LIPS-dedicated equipment currently includes over 40 Unix and Windows-based computers.

7. References

[1] Alchourron, C., P. Gardenfors, and D. Makinson, "On the Logic of Theory Change: Partial Meet Contraction and Revision Functions," *Journal of Symbolic Logic*, pp. 510-530, 1985.

[2] Katsuno, H. and A. O. Mendelzon, "On the Difference between Updating a Knowledge Database and Revising It," presented at Second International Conference on Principles of Knowledge Representation and Reasoning (KR'91), 1991.

[3] Darwiche, A. and J. Pearl, "On the Logic of Iterated Belief Revision,", vol. 89, pp. 1-29, 1997.

[4] Boutilier, C., "Iterated Revision and Minimal Revision of Conditional Beliefs," *Journal of Philosophical Logic*, pp. 263-305, 1996.

[5] Cordier, M. O. and J. Lang, "Linking transitionbased update and base revision," presented at ECSQARU '95, Fribourg, Switzerland, 1995.

[6] Boutilier, C., "Generalized update: belief change in dynamic settings," presented at IJCAI '95, 1995.

[7] Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[8] Dragoni, A. D. and P. Giorgini, "Belief Revision Through the Belief-function Formalism in a Multi-Agents Environment," presented at Intelligent Agents III, Agents Theories, Architectures, and Languages: ECAI'96 Workshop, Budapest, Hungary, 1996.

[9] Duda, R. O. and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, 1973.

[10] Barber, K. S. and J. Kim, "Soft Security: Isolating Unreliable Agents from Society," presented at the Fifth International Workshop on Deception, Fraud, and Trust in Agent Societies at the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS2002), Bologna, Italy, 2002.

[11] Reggia, J. A., Nau, D. S., and Wang, P. 1983. Diagnostic expert systems based on a set covering model. International Journal of Man-Machine Studies 19(3): 437-460.

[12] Neapolitan, R. E. 1990. Probabilistic Reasoning in Expert Systems: Theory and Algorithms. New York: Wiley.

8. Appendix – Computational Code

Additional non-core source code can be found in the ~/scr directories of the installation.

8.1. ITAF Matlab Source Code

```
/*
 * MATLAB Compiler: 3.0
 * Date: Wed Sep 10 13:23:28 2003
 * Arguments: "-B" "macro_default" "-O" "all" "-O" "fold_scalar_mxarrays:on"
 * "-O" "fold_non_scalar_mxarrays:on" "-O" "optimize_integer_for_loops:on" "-O"
 * "array_indexing:on" "-O" "optimize_conditionals:on" "-v" "-g" "-G" "-A"
 * "debugline:on" "-O" "none" "-O" "fold_scalar_mxarrays:off" "-O"
 * "fold_non_scalar_mxarrays:off" "-O" "optimize_integer_for_loops:off" "-O"
 * "array_indexing:off" "-O" "optimize_conditionals:off" "-O" "speculate:off"
 * "-t" "-L" "C" "-W" "lib:liblipsmatlb" "-T" "link:lib" "-h" "doDBRSTTR.m"
 * "libmmfile.mlib"
 */
#include "dodbrsttr.h"
#include "libmatlbm.h"

void InitializeModule_dodbrsttr(void) {
}

void TerminateModule_dodbrsttr(void) {
}

static mxArray * Mdodbrsttr(int nargout_,
                            mxArray * zone_status,
                            mxArray * confidence,
                            mxArray * R);

_mexLocalFunctionTable _local_function_table_dodbrsttr
  = { 0, (mexFunctionTableEntry *)NULL };

/*
 * The function "mlfDodbrsttr" contains the normal interface for the
 * "dodbrsttr" M-function from file
 * "d:\cygwin\home\jisun\lips\development\src\edu\utexas\lips\domainspecific\uci
 * rm\itaf\dodbrsttr.m" (lines 1-183). This function processes any input
 * arguments and passes them to the implementation version of the function,
 * appearing above.
 */
mxArray * mlfDodbrsttr(mxArray * zone_status,
                       mxArray * confidence,
                       mxArray * R) {
  int nargout = 1;
  mxArray * a = NULL;
  mlfEnterNewContext(0, 3, zone_status, confidence, R);
  a = Mdodbrsttr(nargout, zone_status, confidence, R);
  mlfRestorePreviousContext(0, 3, zone_status, confidence, R);
  return mlfReturnValue(a);
```

```
}

/*
 * The function "mlxDodbrsttr" contains the feval interface for the "dodbrsttr"
 * M-function from file
 * "d:\cygwin\home\jisun\lips\development\src\edu\utexas\lips\domainspecific\uci
 * rm\itaf\dodbrsttr.m" (lines 1-183). The feval function calls the
 * implementation version of dodbrsttr through this function. This function
 * processes any input arguments and passes them to the implementation version
 * of the function, appearing above.
 */
void mlxDodbrsttr(int nlhs, mxArray * plhs[], int nrhs, mxArray * prhs[]) {
  mxArray * mprhs[3];
  mxArray * mplhs[1];
  int i;
  if (nlhs > 1) {
    mlfError(
      mxCreateString(
        "Run-time Error: File: dodbrsttr Line: 1 Column:"
        " 1 The function \"dodbrsttr\" was called with m"
        "ore than the declared number of outputs (1)."),
      NULL);
  }
  if (nrhs > 3) {
    mlfError(
      mxCreateString(
        "Run-time Error: File: dodbrsttr Line: 1 Column:"
        " 1 The function \"dodbrsttr\" was called with m"
        "ore than the declared number of inputs (3)."),
      NULL);
  }
  for (i = 0; i < 1; ++i) {
    mplhs[i] = NULL;
  }
  for (i = 0; i < 3 && i < nrhs; ++i) {
    mprhs[i] = prhs[i];
  }
  for (; i < 3; ++i) {
    mprhs[i] = NULL;
  }
  mlfEnterNewContext(0, 3, mprhs[0], mprhs[1], mprhs[2]);
  mplhs[0] = Mdodbrsttr(nlhs, mprhs[0], mprhs[1], mprhs[2]);
  mlfRestorePreviousContext(0, 3, mprhs[0], mprhs[1], mprhs[2]);
  plhs[0] = mplhs[0];
}

/*
 * The function "Mdodbrsttr" is the implementation version of the "dodbrsttr"
 * M-function from file
 * "d:\cygwin\home\jisun\lips\development\src\edu\utexas\lips\domainspecific\uci
 * rm\itaf\dodbrsttr.m" (lines 1-183). It contains the actual compiled code for
 * that M-function. It is a static function and must only be called from one of
 * the interface functions, appearing below.
 */
```

```
/*
 * function a = doDBRSTTR (zone_status, confidence, R)
 */
static mxArray * Mdodbrsttr(int nargout_,
                    mxArray * zone_status,
                    mxArray * confidence,
                    mxArray * R) {
  mclMlineEnterFunction(
    "d:\\cygwin\\home\\jisun\\lips\\development\\src\\edu\\u"
    "texas\\lips\\domainspecific\\ucirm\\itaf\\dodbrsttr.m",
    "dodbrsttr")
  mexLocalFunctionTable save_local_function_table_
    = mclSetCurrentLocalFunctionTable(&_local_function_table_dodbrsttr);
  mxArray * a = NULL;
  mxArray * agent_id = NULL;
  mxArray * r = NULL;
  mxArray * idx = NULL;
  mxArray * val = NULL;
  mxArray * w = NULL;
  mxArray * tmp = NULL;
  mxArray * l = NULL;
  mxArray * P_qf = NULL;
  mxArray * k = NULL;
  mxArray * P_qt = NULL;
  mxArray * tbl = NULL;
  mxArray * nEvidence = NULL;
  mxArray * m = NULL;
  mxArray * statements_rank = NULL;
  mxArray * x = NULL;
  mxArray * v = NULL;
  mxArray * n = NULL;
  mxArray * statements_idx = NULL;
  mxArray * statements = NULL;
  mxArray * value = NULL;
  mxArray * c = NULL;
  mxArray * certainty = NULL;
  mxArray * C = NULL;
  mxArray * S = NULL;
  mxArray * N = NULL;
  mxArray * delta = NULL;
  mclCopyArray(&zone_status);
  mclCopyArray(&confidence);
  mclCopyArray(&R);
  /*
   *
   * delta = 0.05;
   */
  mclMline(3);
  mlfAssign(&delta, mlfScalar(0.05));
  /*
   *
   * N = length(zone_status);                % number of sensors
   */
  mclMline(5);
```

```
mlfAssign(&N, mlfScalar(mclLengthInt(mclVa(zone_status, "zone_status"))));
/*
 *
 * a = zeros(1,2);                  % return value
 */
mclMline(7);
mlfAssign(&a, mlfZeros(mlfScalar(1), mlfScalar(2), NULL));
/*
 * % [zone_status,confidence, reputation]
 * %tid = -1;
 *
 * S = [zone_status'];
 */
mclMline(11);
mlfAssign(&S, mlfCtranspose(mclVa(zone_status, "zone_status")));
/*
 * C = 1;                           % # of coordinate
 */
mclMline(12);
mlfAssign(&C, mlfScalar(1));
/*
 *
 *
 * % Show Control variables
 * %disp(['Number of sensors  = ' int2str(N)]);
 * %disp(['Sensor reading = ']);
 * %disp(S);
 * %disp(['The current Reputation = ']);
 * %disp(R);
 *
 * % for each coordinate
 * %================================================================
 *
 * certainty = confidence;
 */
mclMline(25);
mlfAssign(&certainty, mclVa(confidence, "confidence"));
/*
 *
 * for c = 1:C
 */
mclMline(27);
{
  mclForLoopIterator viter__;
  for (mclForStart(&viter__, mlfScalar(1), mclVv(C, "C"), NULL);
       mclForNext(&viter__, &c);
       ) {
    /*
     * %    disp(['For sensor[' int2str(c) ']...']);
     * %  clear statements statements_idx;
     * value           = S(:,c);          % 1 x N vector
     */
    mclMline(30);
    mlfAssign(
```

```
 &value,
 mlfIndexRef(
  mclVv(S, "S"), "(?,?)", mlfCreateColonIndex(), mclVv(c, "c")));
/*
 * statements       = value(1);        % the first perceived value
 */
mclMline(31);
mlfAssign(
 &statements,
 mlfIndexRef(mclVv(value, "value"), "(?)", mlfScalar(1)));
/*
 * statements_idx{1} = 1;
 */
mclMline(32);
mlfIndexAssign(&statements_idx, "{?}", mlfScalar(1), mlfScalar(1));
/*
 * for n = 2:N                        % for the rest of sensors
 */
mclMline(33);
{
  mclForLoopIterator viter__0;
  for (mclForStart(&viter__0, mlfScalar(2), mclVv(N, "N"), NULL);
       mclForNext(&viter__0, &n);
       ) {
    /*
     * v = value(n);
     */
    mclMline(34);
    mlfAssign(
     &v,
     mlfIndexRef(mclVv(value, "value"), "(?)", mclVv(n, "n")));
    /*
     * x = find(statements == v);         % new statement?
     */
    mclMline(35);
    mlfAssign(
     &x,
     mlfFind(
       NULL,
       NULL,
       mclEq(mclVv(statements, "statements"), mclVv(v, "v"))));
    /*
     * if isempty(x)                      % if new statement
     */
    mclMline(36);
    if (mlfTobool(mlfIsempty(mclVv(x, "x")))) {
      /*
       * statements     = [statements v];  % then record it
       */
      mclMline(37);
      mlfAssign(
       &statements,
       mlfHorzcat(
         mclVv(statements, "statements"),
```

```
            mclVv(v, "v"),
            NULL));
          /*
           * statements_idx{length(statements)} = n;
           */
          mclMline(38);
          mlfIndexAssign(
           &statements_idx,
           "{?}",
           mlfScalar(
            mclLengthInt(mclVv(statements, "statements"))),
           mclVv(n, "n"));
        /*
         * else                    % if existing
         */
        mclMline(39);
        } else {
          /*
           * statements_idx{x} = [statements_idx{x} n];
           */
          mclMline(40);
          mlfIndexAssign(
           &statements_idx,
           "{?}",
           mclVv(x, "x"),
           mlfHorzcat(
            mlfIndexRef(
             mclVv(statements_idx, "statements_idx"),
             "{?}",
             mclVv(x, "x")),
            mclVv(n, "n"),
            NULL));
        /*
         * end
         */
        mclMline(41);
        }
      /*
       * end
       */
      mclMline(42);
      }
      mclDestroyForLoopIterator(viter__0);
}
/*
 *
 * statements_rank = [];
 */
mclMline(44);
mlfAssign(&statements_rank, mclCreateEmptyArray());
/*
 * for m = 1:length(statements)  % for each statement
 */
mclMline(45);
```

```
{
  mclForLoopIterator viter__1;
  for (mclForStart(
       &viter__1,
       mlfScalar(1),
       mlfScalar(mclLengthInt(mclVv(statements, "statements"))),
       NULL);
      mclForNext(&viter__1, &m);
      ) {
    /*
     * nEvidence = length(statements_idx{m});
     */
    mclMline(46);
    mlfAssign(
     &nEvidence,
     mclFeval(
       mclValueVarargout(),
       mlxLength,
       mlfIndexRef(
         mclVv(statements_idx, "statements_idx"),
         "{?}",
         mclVv(m, "m")),
       NULL));
    /*
     *
     * %disp(['statement m = [' int2str(m) ' / ' ...
     * %      int2str(length(statements)) 'sets] supported by (' ...
     * %      int2str(nEvidence) ') ' ]);
     * %disp(statements_idx{m});
     * %disp(['value = ' num2str(statements(m))]);
     * %for k = 1:length(statements_idx{m})
     * %  disp(['Reputation of ' ...
     * %        sprintf('%d is %f', ...
     * %              statements_idx{m}(k), ...
     * %              R(statements_idx{m}(k)))
     * %       ]);
     * %    end
     * % disp(['SIZE OF nEvidence = ' num2str(nEvidence)]);
     *
     * tbl = zeros(1,nEvidence);
     */
    mclMline(62);
    mlfAssign(
     &tbl,
     mlfZeros(
       mlfScalar(1), mclVv(nEvidence, "nEvidence"), NULL));
    /*
     * % for large nEvidence %
     * % if nEvidence <=53
     * for n = 1:pow2(nEvidence)-1
     */
    mclMline(65);
    {
      mclForLoopIterator viter__2;
```

```
  for (mclForStart(
        &viter__2,
        mlfScalar(1),
        mclMinus(
         mlfPow2(mclVv(nEvidence, "nEvidence"), NULL),
         mlfScalar(1)),
        NULL);
       mclForNext(&viter__2, &n);
       ) {
      /*
       * tbl = [tbl; bitget(n,nEvidence:-1:1)];
       */
      mclMline(66);
      mlfAssign(
       &tbl,
       mlfVertcat(
        mclVv(tbl, "tbl"),
        mlfBitget(
         mclVv(n, "n"),
         mlfColon(
          mclVv(nEvidence, "nEvidence"),
          mlfScalar(-1),
          mlfScalar(1))),
        NULL));
    /*
     * end
     */
    mclMline(67);
    }
    mclDestroyForLoopIterator(viter__2);
}
/*
 * %end
 *
 * %tbl = zeros(pow2(nEvidence)-1, nEvidence);
 *
 * %[row_SIZE, col_SIZE] = size(tbl);
 *
 * %for zx=1:col_SIZE
 * %    col_x = col_SIZE - zx;    % 0 to
 * %    num01 = pow2(col_x);
 * %    num_1 = 0;
 * %    num_0 = 1;
 * %    for zy = 1:row_SIZE
 * %       if (num_1 == 0)
 * %          tbl(zy,zx) = 0;
 * %          num_0 = num_0 + 1;
 * %       end
 * %       if (num_0 == 0)
 * %          tbl(zy,zx) = 1;
 * %          num_1 = num_1 + 1;
 * %       end
 * %
 * %        if (num_0 > pow2(col_x))
```

```
* %           num_0 = 0;
* %           num_1 = 1;
* %         end
* %
* %         if (num_1 > pow2(col_x))
* %               num_0 = 1;
* %               num_1 = 0;
* %         end
* %       end
* %   end
*
*
*
*
*
*
*
*
*
*
*
* % Calc P(q^t)
* P_qt = 1.0;
*/
mclMline(112);
mlfAssign(&P_qt, mlfScalar(1));
/*
*
* for k = 1:nEvidence              % for each agent supporting statement m
*/
mclMline(114);
{
  mclForLoopIterator viter__3;
  for (mclForStart(
       &viter__3,
       mlfScalar(1),
       mclVv(nEvidence, "nEvidence"),
       NULL);
     mclForNext(&viter__3, &k);
     ) {
     /*
      * P_qt = P_qt * certainty(statements_idx{m}(k)) * ...
      */
     mclMline(115);
     mlfAssign(
      &P_qt,
      mclMtimes(
       mclMtimes(
        mclVv(P_qt, "P_qt"),
        mlfIndexRef(
         mclVv(certainty, "certainty"),
         "(?)",
         mlfIndexRef(
          mclVv(statements_idx, "statements_idx"),
```

```
            "{?}(?)",
            mclVv(m, "m"),
            mclVv(k, "k")))),
         mlfIndexRef(
          mclVa(R, "R"),
          "(?)",
          mlfIndexRef(
           mclVv(statements_idx, "statements_idx"),
           "{?}(?)",
           mclVv(m, "m"),
           mclVv(k, "k")))));
  /*
   * R(statements_idx{m}(k));
   * end
   */
  mclMline(117);
  }
  mclDestroyForLoopIterator(viter__3);
}
/*
 *
 * % Calc P(q^f)
 * P_qf = 0.0;
 */
mclMline(120);
mlfAssign(&P_qf, mlfScalar(0));
/*
 *
 * for l = 1:pow2(length(statements_idx{m}))
 */
mclMline(122);
{
  mclForLoopIterator viter__4;
  for (mclForStart(
       &viter__4,
       mlfScalar(1),
       mlfPow2(
        mclFeval(
         mclValueVarargout(),
         mlxLength,
         mlfIndexRef(
          mclVv(statements_idx, "statements_idx"),
          "{?}",
          mclVv(m, "m")),
         NULL),
        NULL),
       NULL);
      mclForNext(&viter__4, &l);
      ) {
     /*
      *
      * tmp = 1.0;
      */
     mclMline(124);
```

```
mlfAssign(&tmp, mlfScalar(1));
/*
 *
 * for k = 1:length(statements_idx{m})
 */
mclMline(126);
{
  mclForLoopIterator viter__5;
  for (mclForStart(
        &viter__5,
        mlfScalar(1),
        mclFeval(
          mclValueVarargout(),
          mlxLength,
          mlfIndexRef(
            mclVv(
              statements_idx, "statements_idx"),
            "{?}",
            mclVv(m, "m")),
          NULL),
        NULL);
       mclForNext(&viter__5, &k);
       ) {
      /*
       *
       *
       * if tbl(l,k) == 0
       */
      mclMline(129);
      if (mlfTobool(
            mclEq(
              mlfIndexRef(
                mclVv(tbl, "tbl"),
                "(?,?)",
                mclVv(l, "l"),
                mclVv(k, "k")),
              mlfScalar(0)))) {
        /*
         * %if my_val_1_k == 0
         * tmp = tmp * (1-certainty(statements_idx{m}(k))) * ...
         */
        mclMline(131);
        mlfAssign(
          &tmp,
          mclMtimes(
            mclMtimes(
              mclVv(tmp, "tmp"),
              mclMinus(
                mlfScalar(1),
                mlfIndexRef(
                  mclVv(certainty, "certainty"),
                  "(?)",
                  mlfIndexRef(
                    mclVv(
```

```
                    statements_idx,
                    "statements_idx"),
                  "{?}(?)",
                  mclVv(m, "m"),
                  mclVv(k, "k")))),
              mlfIndexRef(
               mclVa(R, "R"),
               "(?)",
               mlfIndexRef(
                mclVv(
                 statements_idx,
                 "statements_idx"),
                "{?}(?)",
                mclVv(m, "m"),
                mclVv(k, "k")))));
      /*
       * R(statements_idx{m}(k));
       * else
       */
      mclMline(133);
      } else {
        /*
         * tmp = tmp * (1-R(statements_idx{m}(k)));
         */
        mclMline(134);
        mlfAssign(
         &tmp,
         mclMtimes(
          mclVv(tmp, "tmp"),
          mclMinus(
           mlfScalar(1),
           mlfIndexRef(
            mclVa(R, "R"),
            "(?)",
            mlfIndexRef(
             mclVv(
              statements_idx,
              "statements_idx"),
             "{?}(?)",
             mclVv(m, "m"),
             mclVv(k, "k"))))));
      /*
       * end
       */
      mclMline(135);
      }
    /*
     * end
     */
    mclMline(136);
    }
    mclDestroyForLoopIterator(viter__5);
}
/*
```

```
        * P_qf = P_qf + tmp;
        */
       mclMline(137);
       mlfAssign(
        &P_qf,
        mclPlus(mclVv(P_qf, "P_qf"), mclVv(tmp, "tmp")));
     /*
      * end
      */
     mclMline(138);
     }
     mclDestroyForLoopIterator(viter__4);
   }
   /*
    *
    * P_qt = sqrt(P_qt);
    */
   mclMline(140);
   mlfAssign(&P_qt, mlfSqrt(mclVv(P_qt, "P_qt")));
   /*
    * P_qf = sqrt(P_qf);
    */
   mclMline(141);
   mlfAssign(&P_qf, mlfSqrt(mclVv(P_qf, "P_qf")));
   /*
    *
    * %   disp(['P(q^t) = ' num2str(P_qt)]);
    * %   disp(['P(q^f) = ' num2str(P_qf)]);
    *
    * w = inv(P_qt+P_qf);
    */
   mclMline(146);
   mlfAssign(
    &w,
    mlfInv(
     mclPlus(mclVv(P_qt, "P_qt"), mclVv(P_qf, "P_qf"))));
   /*
    * statements_rank(m) = w * P_qt;
    */
   mclMline(147);
   mlfIndexAssign(
    &statements_rank,
    "(?)",
    mclVv(m, "m"),
    mclMtimes(mclVv(w, "w"), mclVv(P_qt, "P_qt")));
/*
 *
 * %   disp(['E{P(q^t)} = ' num2str(statements_rank(m))]);
 *
 * end
 */
mclMline(151);
}
mclDestroyForLoopIterator(viter__1);
```

```
}
/*
 *
 * [val,idx] = max(statements_rank);    % winner
 */
mclMline(153);
mlfAssign(
 &val,
 mlfMax(
  &idx, mclVv(statements_rank, "statements_rank"), NULL, NULL));
/*
 *
 * a(c*2-1) = statements(idx);          % belief value
 */
mclMline(155);
mlfIndexAssign(
 &a,
 "(?)",
 mclMinus(mclMtimes(mclVv(c, "c"), mlfScalar(2)), mlfScalar(1)),
 mlfIndexRef(
  mclVv(statements, "statements"), "(?)", mclVv(idx, "idx")));
/*
 * a(c*2)   = val;                      % belief certainty
 */
mclMline(156);
mlfIndexAssign(
 &a,
 "(?)",
 mclMtimes(mclVv(c, "c"), mlfScalar(2)),
 mclVv(val, "val"));
/*
 *
 * %  disp(['Winner is ' num2str(a(c*2-1))]);
 * %  disp(['   with certainty = ' num2str(val)]);
 *
 *
 * %revises reputations
 * for m = 1:length(statements)  % for each statement
 */
mclMline(163);
{
  mclForLoopIterator viter__6;
  for (mclForStart(
       &viter__6,
       mlfScalar(1),
       mlfScalar(mclLengthInt(mclVv(statements, "statements"))),
       NULL);
     mclForNext(&viter__6, &m);
     ) {
    /*
     * nEvidence = length(statements_idx{m});
     */
    mclMline(164);
    mlfAssign(
```

```
 &nEvidence,
 mclFeval(
  mclValueVarargout(),
  mlxLength,
  mlfIndexRef(
   mclVv(statements_idx, "statements_idx"),
   "{?}",
   mclVv(m, "m")),
  NULL));
/*
 * if m == idx                    % if winner bin
 */
mclMline(165);
if (mlfTobool(mclEq(mclVv(m, "m"), mclVv(idx, "idx")))) {
  mclForLoopIterator viter__7;
  /*
   *
   * for r = 1 : nEvidence
   */
  mclMline(167);
  for (mclForStart(
       &viter__7,
       mlfScalar(1),
       mclVv(nEvidence, "nEvidence"),
       NULL);
     mclForNext(&viter__7, &r);
     ) {
    /*
     * agent_id = statements_idx{m}(r);
     */
    mclMline(168);
    mlfAssign(
     &agent_id,
     mlfIndexRef(
      mclVv(statements_idx, "statements_idx"),
      "{?}(?)",
      mclVv(m, "m"),
      mclVv(r, "r")));
    /*
     * R(agent_id) = R(agent_id) + delta*(1.0-R(agent_id));
     */
    mclMline(169);
    mlfIndexAssign(
     &R,
     "(?)",
     mclVv(agent_id, "agent_id"),
     mclPlus(
      mlfIndexRef(
       mclVa(R, "R"),
       "(?)",
       mclVv(agent_id, "agent_id")),
      mclMtimes(
       mclVv(delta, "delta"),
       mclMinus(
```

```
            mlfScalar(1),
            mlfIndexRef(
              mclVa(R, "R"),
              "(?)",
              mclVv(agent_id, "agent_id"))))));
    /*
     * %      disp(['+: ' num2str(agent_id) ', to ' num2str(R(agent_id))]);
     * end
     */
    mclMline(171);
    }
    mclDestroyForLoopIterator(viter__7);
/*
 * else                         % not in the winner bin
 */
mclMline(172);
} else {
    mclForLoopIterator viter__8;
    /*
     * for r = 1 : nEvidence
     */
    mclMline(173);
    for (mclForStart(
          &viter__8,
          mlfScalar(1),
          mclVv(nEvidence, "nEvidence"),
          NULL);
        mclForNext(&viter__8, &r);
        ) {
        /*
         * agent_id = statements_idx{m}(r);
         */
        mclMline(174);
        mlfAssign(
          &agent_id,
          mlfIndexRef(
            mclVv(statements_idx, "statements_idx"),
            "{?}(?)",
            mclVv(m, "m"),
            mclVv(r, "r")));
        /*
         * R(agent_id) = R(agent_id) - delta*R(agent_id);
         */
        mclMline(175);
        mlfIndexAssign(
          &R,
          "(?)",
          mclVv(agent_id, "agent_id"),
          mclMinus(
            mlfIndexRef(
              mclVa(R, "R"),
              "(?)",
              mclVv(agent_id, "agent_id")),
            mclMtimes(
```

```
                        mclVv(delta, "delta"),
                        mlfIndexRef(
                          mclVa(R, "R"),
                          "(?)",
                          mclVv(agent_id, "agent_id")))));
                    /*
                     * %     disp(['-: ' num2str(agent_id) ', to ' num2str(R(agent_id))]);
                     * end
                     */
                    mclMline(177);
                    }
                    mclDestroyForLoopIterator(viter__8);
                  /*
                   * end
                   */
                  mclMline(178);
                  }
                /*
                 * end
                 */
                mclMline(179);
                }
                mclDestroyForLoopIterator(viter__6);
              }
            /*
             *
             * end
             */
            mclMline(181);
            }
            mclDestroyForLoopIterator(viter__);
          }
          /*
           * a = [a R];
           */
          mclMline(182);
          mlfAssign(&a, mlfHorzcat(mclVv(a, "a"), mclVa(R, "R"), NULL));
          mclValidateOutput(a, 1, nargout_, "a", "dodbrsttr");
          mxDestroyArray(delta);
          mxDestroyArray(N);
          mxDestroyArray(S);
          mxDestroyArray(C);
          mxDestroyArray(certainty);
          mxDestroyArray(c);
          mxDestroyArray(value);
          mxDestroyArray(statements);
          mxDestroyArray(statements_idx);
          mxDestroyArray(n);
          mxDestroyArray(v);
          mxDestroyArray(x);
          mxDestroyArray(statements_rank);
          mxDestroyArray(m);
          mxDestroyArray(nEvidence);
          mxDestroyArray(tbl);
```

```
    mxDestroyArray(P_qt);
    mxDestroyArray(k);
    mxDestroyArray(P_qf);
    mxDestroyArray(l);
    mxDestroyArray(tmp);
    mxDestroyArray(w);
    mxDestroyArray(val);
    mxDestroyArray(idx);
    mxDestroyArray(r);
    mxDestroyArray(agent_id);
    mxDestroyArray(R);
    mxDestroyArray(confidence);
    mxDestroyArray(zone_status);
    mclSetCurrentLocalFunctionTable(save_local_function_table_);
    mclMlineFunctionReturn()
    return a;
    mclMlineExitFunctionReturn();
}
```

8.2. EM Source Code

The source code for the Extend model is within the Extend application and can be viewed through Extend.

9. List of Symbols, Abbreviations, and Acronyms

ACAD	Automatic Chemical Agent Alarm
ACTD	Advanced Concept Technology Demo
AFB	Air Force Base
AFOSR	Air Force Office of Sponsored Research
AFRL	Air Force Research Laboratory
AGM	Belief revision approach named after the authors: Alchourron, Gardenfors, and Makinson
AI	Artificial Intelligence
AODB	Air Operations Database
ASIV	Agent Supported Information Visualization
ATM	Assumption-based Truth Maintenance
ATO	Air Tasking Order
Bayesian belief nets	Belief networks that use probability theory to manage uncertainty by explicitly representing the conditional dependencies between different knowledge components
BCIRM	Biological/Chemical Incident Response Monitor
C^2	Command and Control
CASPOD	Contamination Avoidance at Seaports of Debarkation
CBIRT	Chemical/Biological Incident Response Tool
CBRN	Chemical, Biological, Radiological, or Nuclear
CORBA	Common Object Request Broker Architecture
DAG	Directed Acyclic Graph
Dempster-Shafer	Belief revision that follows the Dempster-Shafer Theory of Evidence: the notion of belief function as adequate representation of one's degrees of belief and the reasoning mechanism based on the rule of combination
DPRK	Democratic People's Republic of (North) Korea
DTRA	Defense Threat Reduction Agency
EM	Effects Model
Expert System	Computer software that can solve a narrowly defined set of problems using information and reasoning techniques normally associated with a human expert
FLYOPS	Flight Operations
GCSS	Global Combat Support System
HAZMAT	Hazardous Materials
GIS	Geographical Information System
GU	Generalized Update belief revision approach
IR	Iterated Revision belief revision approach
ITAF	Information Trustworthiness Assessment Facility
ITWG	Information Technology Working Group
JAOC	Joint Air Operations Center
JOEF	The Joint Operational Effects Federation (JOEF) project
K	Knowledge Base
KB	Knowledge Background

KM	Belief revision approach named after the authors: Katsuno and Mendelzon
MAS	Multi-Agent System
MOPP	Mission Oriented Protective Posture
NBC	Nuclear/Biological/Chemical
PPE	Personal Protective Equipment
Poly-tree	A singly connected network where the underlying undirected graph has no more than one path between any two nodes
RESTOPS	Restoration of Operations
SA	UT:LIPS Sensible Agent intelligent agent architecture
SBIR	Small Business Innovation Research Program
SEP	Scenario Engineering Process
SRC	Survival Recovery Center
STTR	Small Business Technology Transfer
TBM	Tactical Ballistic Missile
TBMCS-UL	Theater Battle Management Core Systems – Unit Level
TBU	Transition-Based Update belief revision approach
UAI	Uncertainty in Artificial Intelligence
UCIRM	User Configurable Incident Response Monitor
UI	User Interface
UT:LIPS	Laboratory for Intelligent Processes and Systems at the University of Texas at Austin
WCCS	Wing Command and Control System
WMD	Weapons of Mass Destruction
WOC	Wing Operations Center